REVISTA PARA USUARIOS DE

Drean Commodore

DICIEMBRE 1985 REP. ARGENTINA Nº 1 ₳ 2.00



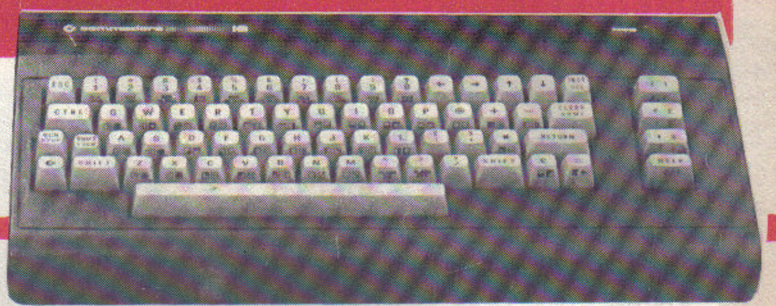
SUMARIO

NOTAS TECNICAS

PROGRAMAS

REVISION DE SOFTWARE

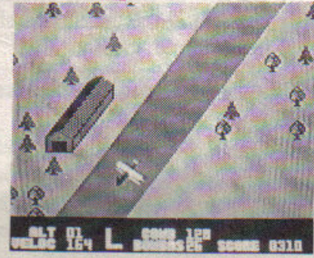
SECCIONES FIJAS

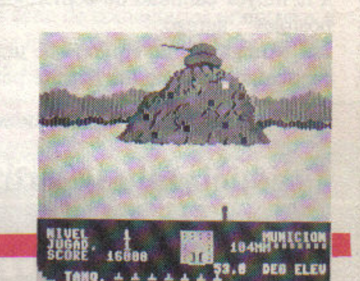


Presentamos la C-16, computadora avanzada con características que sólo se encuentran en equipos de precios muy superiores



Primeros pasos para teclear la C-64





comentamos los programas que se encuentran en el mercado argentino

Drean Commodore AÑO 1 Nº 1 DICIEMBRE DE 1985

Director General
Ernesto del Castillo
Director Editorial
Cristian Pusso
Director Periodístico
Fernando Flores
Director Financiero
Javier Campos Malbrán

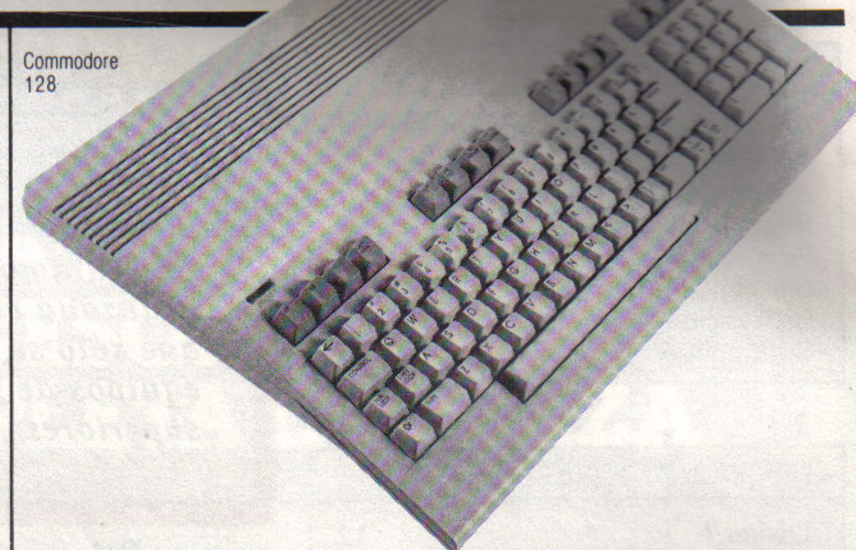
Asesor Editorial
Armengol Torres Sabaté
Arte y Diagramación
Fernando Amengual
Coordinador
Ariel Testori
Secretaria
Moni Ocampo

Gerente de Circulación
Guillermo González Aldalur
Departamento de Publicidad
Jefe: Dolores Urien
Promotora: Marita González
Departamento de Avisos
Oscar Devoto

Drean Commodore es una Revista mensual editada por Editorial PROEDI S.A. (e./f.), Cerrito 1320, 1º Piso, Buenos Aires, Te.: 42-9681/9. Registro Nacional de la Propiedad Intelectual: E.T. M. registrada.

Precio de este ejemplar: ₳ 2.
Impresión: Calcotam. Fotocromo tapa: Columbia. Fotocomposición: Van Waveren.
Los ejemplares atrasados se venderán al precio del último número en circulación.

120.000 C16 y C64



Commodore 128

La Commodore 64 y la 16 se están vendiendo en todos los negocios especializados y en los ya tradicionales de artículos para el hogar. Tal el caso de Frávega, Ozores, Hogar Obrero, etc. Drean no sólo va a trabajar la C-16 y C 64, sino que muy pronto incorporará la 128 y la "Amiga", el último modelo de Commodore que ha salido en Estados Unidos. La idea de la empresa es tratar de venderlos en forma masiva. "Como sucedió con otros productos, queremos lograr el liderazgo en la venta de computadoras. Pensamos que con una marca como Commodore y el respaldo y prestigio de Drean se pueden unir los esfuerzos y llegar a ser líderes en el mercado de las Home Computer", dijo Pedro Nemesio, gerente de Ventas de la División Informática de Drean.
"A pesar de que hace muy poco que salimos a la venta hemos logrado una aceptación muy grande tratando de llegar a la mayor cantidad de comercios posibles", agregó.
La "Amiga" es un equipo que tiene una capacidad bastante interesante para trabajar y muchas posibilidades de expansión. Puede llegar a trabajar con disco rígido, tiene gran variedad de colores, sonido estereofónico y pantalla tridimensional, pudiendo reproducir gráficos de excelente calidad. Esperan lanzarla al mercado para el primer trimestre de 1986.

La C-16 y la C-64 ya se están fabricando en la planta de Drean de San Luis. Para 1986 se está encarando una producción que totalizará entre 100 y las 120 mil máquinas.
Para fines de diciembre, la línea podría completarse agregando a la consola, el drive de diskette, dataset y una impresora.

Producirán un Cartridge

Peek está tratando de pasar a casi cualquier programa de hasta 32K de memoria a un cartridge. Esto se hace mediante una parte de soft nueva y una parte de hard, con lo cual a casi cualquier programa lo podemos pasar a un cartridge. Con el mismo se tiene la facilidad de instalarlo en segundos, no se depende de un drive ni del dataset, ahorrando mucho tiempo.
Entre las características técnicas resalta la capacidad de memoria de 32 K. En este momento se están desarrollando las técnicas necesarias para tener un 100% de seguridad en todos los programas que se están llevando a cartridge.
El material se trajo de Estados Unidos y ya se están haciendo las pruebas de desarrollo. Se calcula que para fines de diciembre se empezarán a producir los primeros modelos que luego serán copiados en cantidad.
Habiéndose ampliado la capacidad de almacenamiento de 16 a 32 Kbytes, también se han aumentado considerablemente las posibilidades del cartridge frente a otros medios (drive, cassette).

Administrador del Disk Drive 1541

Acaba de ingresar al mercado un programa Administrador de la 1541 - 1541 Disk Manager. El mismo mantiene el diskette continuamente organizado, envía más comandos al drive, realiza copias de archivos y back-up, puede borrar varios archivos simultáneamente, e imprime la dirección inicial de los programas escritos en lenguaje máquina (generalmente esta dirección está oculta y no es posible determinarla).

COMMODORE NEWS

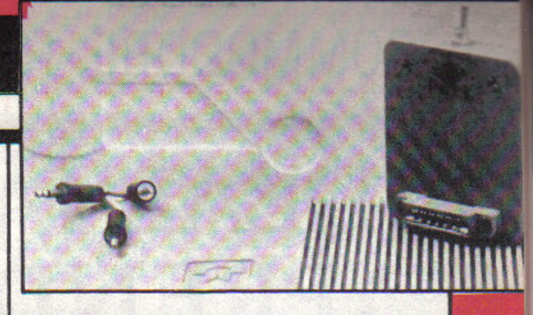
Controlador de voz

El sistema controlador de voz para la C-64 puede almacenar en diskettes la voz del usuario codificada en pulsos digitales. Cada una de estas grabaciones pueden contener un total de 64 sonidos, palabras o frases. El sistema es capaz de reconocer exactamente la voz original. De esta forma se puede, por ejemplo, usar la propia voz para acceder a información confidencial. Es decir, que actuaría como "clave" para que ningún extraño pueda entrar en la base de datos.

Nueva interface para grabador

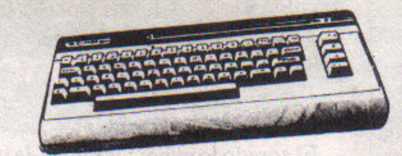
PEEK acaba de lanzar al mercado argentino una nueva interfase para el equipo C-64. Esta le permite al usuario cargar y leer programas utilizando un grabador común sin ser necesario el Datasette. Su instalación es muy sencilla y no se requieren otros elementos salvo los que PEEK suministra.
Antes de conectarla al equipo se debe comprobar que éste se encuentre apagado. Luego se conecta la cinta plana al Port destinado al uso del Datasette. Paso seguido hay que conectar los cables que acompañan a la Interfase al grabador. Uno de ellos es de color rojo y se debe insertar en la entrada MIC del grabador. El otro cable es de color negro y debe ir a la entrada EAR.
Luego se puede, por ejemplo, cargar un programa desde el cassette. Esto se realiza poniendo el volumen entre la mitad y el máximo, y colocando el control de tonos al máximo de agudos.
Para grabar los programas simplemente debemos ejecutar el comando SAVE presionando la tecla de grabación correspondiente.
Con la Interfase, PEEK suministra una extensa y clara documentación en la cual se explica en forma detallada la conexión y el funcionamiento de la misma. Además de una serie de comentarios orientados a resolver los problemas que se presenten durante la grabación/lectura.

INTERFASE PARA GRABADOR



Lt. KERNAL - Sistema de disco duro

Fue lanzado para la C-64 y para la 128. Es un dispositivo de almacenamiento masivo de información. Su capacidad oscila entre los 10 y 20 Mbytes. Internamente es administrado por el sistema operativo D.O.S., el cual tiene una longitud de 140 Kbytes (que reside en el sistema), siendo totalmente compatible con el sistema operativo de la 1541. Este consiste en un controlador de disco duro. Se conecta en el port de cartridge de la C-64. La velocidad de carga es muy alta. Por ejemplo, el tiempo que tarda en almacenar 16 Kbytes es de, aproximadamente, un segundo. Provee también nuevos comandos orientados a una mejor administración del recurso. Además, permite que la 1541 esté conectada a la C-64 pudiendo trabajar simultáneamente.





IDEAL PARA PROGRAMAR, APRENDER Y JUGAR

La Commodore 16 es una computadora doméstica muy avanzada con características que, normalmente, sólo se encuentran en equipos de un precio muy superior. Fue diseñada pensando en que sea fácil de programar y de utilizar.

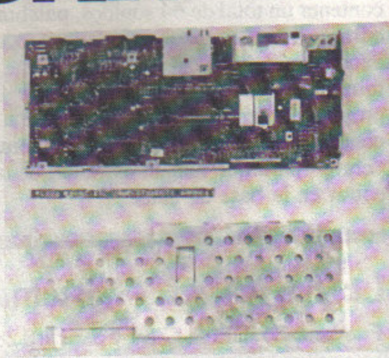
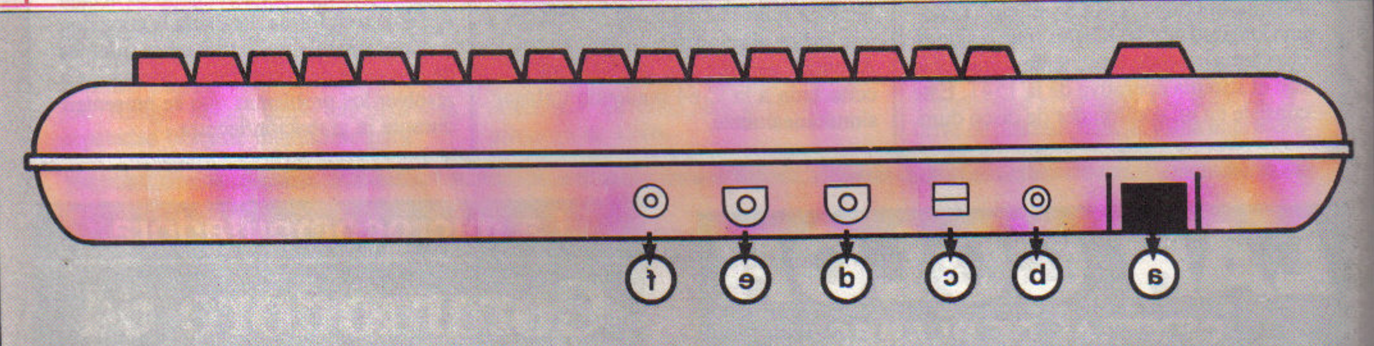


Figura 1

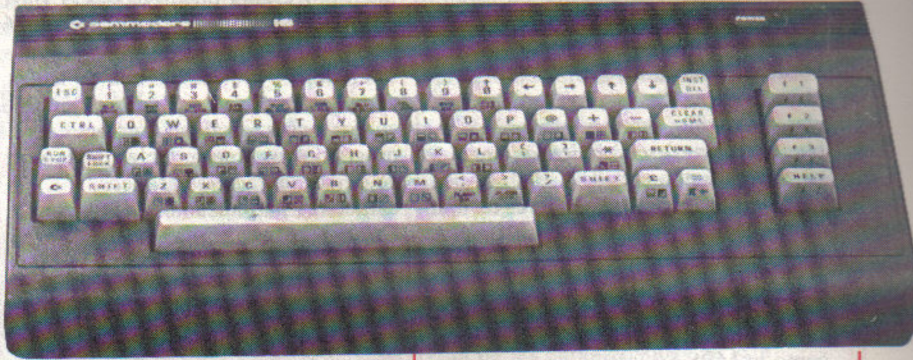


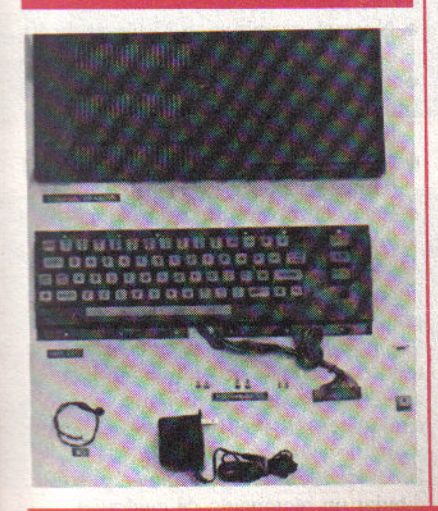
a) Salida cartucho b) Salida a televisor c) Modulador d) Salida a monitor e) Port serial periféricos f) port cassette

Comparación entre Commodore 16 y Spectrum

	SPECTRUM	COMMODORE 16
Colores	8 col. x 2 tonos = 16 16 ó 48 KBytes	15 col, x 8 tonos = 121 16 K Bytes
Lenguaje	Basic	Basic 3,5 es una versión mejorada del modelo 64
Pantalla	La impresión en pantalla se puede hacer con letras en colores (8), fondo en color (8), brillo (2 tonos) y opción a que los colores parpadeen intermitentemente. 90 comandos Teclado de goma Sin teclas programables.	La impresión es similar , con más colores y posibilidades de tonos. Los colores no pueden parpadear. 110 comandos Teclado de plástico tipo máquina de escribir. 7 teclas con funciones programables.
Sonido	El sonido lo genera a través de un altavoz incorporado con un sonido muy débil. Con una gama de 10 octavas de sonido.	El sonido lo genera a través del televisor con la posibilidad de regular el sonido.
Conclusión	La ventaja que presenta es la posibilidad de adquirir con 48 K Bytes de memoria. Y la desventaja más grande se presenta en su teclado de goma debido a que éste hace que sus funciones sean más lentas y con mayor posibilidad de error.	Las posibilidades de uso y sus funciones la asemejan a equipo profesional, con la diferencia en la capacidad de memoria y su facilidad de manejo.

La C-16 está dotado de un teclado profesional que tiene fama de envidiable dado que es el mismo que equipa al modelo de la Comodore 64.
La Commodore 16 es una máquina ideal, por los nuevos elementos que presenta, para aprender a desarrollar programas.
La educación es un área donde esta nueva computadora encaja perfectamente con programas que van desde el pre-escolar hasta cualquier nivel del proceso educativo.
El aprendizaje de la informática es muy importante en nuestros días y la Commodore 16 es una herramienta que ayuda a familiarizarse y mantenerse en contacto con una tecnología que crece día a día.
Esta nueva máquina ofrece una opción insuperable en programación, educación y juegos, para entrar en el mundo de la informática. Hecha la presentación sólo falta conocerla en sus aspectos técnicos.

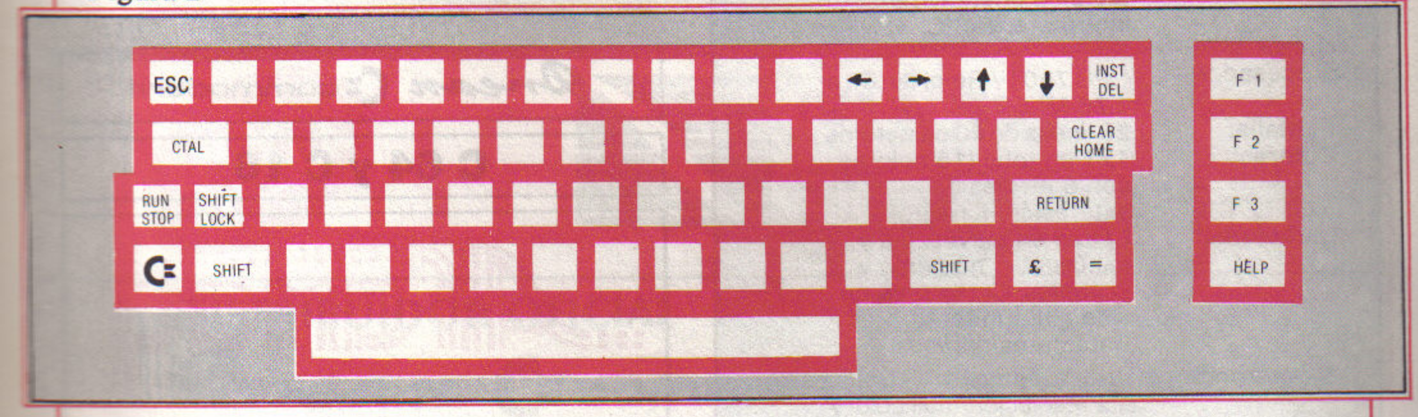




Capacidad del equipo.

Para resaltar las cualidades del equipo se debe remitir al estudio de los elementos componentes del hardware y software. La Commodore 16 ofrece cinco elementos indispensables para aplicaciones a nivel recreativo, hogareño y educativo.
1°) Imágenes con gráficos en modo normal y alta resolución, permitiendo la fácil graficación de funciones trigonométricas tales como seno,

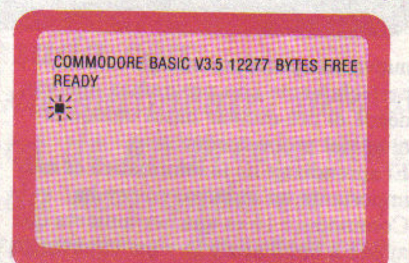
Figura 2



coseno, etc.
2°) 121 posibilidades de color dadas por 15 colores básicos (rojo, rosa, bordeaux (púrpura), azul oscuro, azul claro, azul, verde oscuro, verde, verde claro, amarillo, marrón, anaranjado, blanco y negro), multiplicados por 8 niveles de luminosidad.
3°) Sonido con dos generadores de tono, o uno de tono y otro de ruido blanco. Posee 8 niveles de volumen.
4°) Compatibilidad con todos los periféricos de la línea Commodore 64, por ejemplo, el Drive 1541 o impresoras como la MPS 803/802, etc.
5°) Fácil comprensión y manejo del BASIC.

Configuración técnica.

C.P.U. La memoria RAM es de 16 K total, donde 12 K son los realmente libres para disponer de un programa en lenguaje BASIC. La memoria RAM ocupa 32K, incluyendo el sistema operativo y el intérprete BASIC. El microprocesador que utiliza es el 7501 con dos velocidades de hora (.89 x 1.76 MHz.)
Teclado: Es profesional con 66 teclas, compuestas por un teclado numérico, cuatro teclas con ocho funciones programables, cuatro teclas de control de cursor, caracteres gráficos, caracteres parpadeantes, ESC, etc.
Conexiones de input/output. Posee port serial para cartridge y paralelo a Disk Drive, dos puertos para Joystick, puerto para interfase de Datasette, salida de radiofrecuencia, canal 3 ó 4, port de salida de video, entrada y salida de audio y entrada de suministro de energía.



Capacidad Gráfica

El equipo ofrece cuatro modos de resolución en pantalla:

- Normal, permitiendo 25 líneas de 40 caracteres cada una.
- Alta resolución, con una definición de 320 puntos por línea (total, 200 líneas).
- Multicolor, con una definición de 160 puntos x 200.
- Pantalla mixta, alta resolución y texto.

Accionando determinados comandos se pueden lograr especificaciones en los gráficos. Por ejemplo:
Circle: este comando permite dibujar todo o una parte de un polígono, en cualquier parte de la pantalla usando cualquieer otro color x, y radio, rotación y ángulo entre lados.
Box: usando este comando se puede dibujar un rectángulo de cualquier medida, en cúalquier lugar de la

Ficha Técnica

Memoria Ram:	16 K en total - 12 Klibres para programas en BASIC
Memoria Rom:	32 K ROM (incluído Sistema operativo e intérprete BASIC)
Micro-procesador:	CPU 7501 / Reloj de .89 a 1.76 Mhz
Pantalla:	25 líneas de 40 caracteres
Colores:	121 en total (15 colores con 8 niveles de saturación + negro)
Caracteres:	Letras en mayúscula y minúscula - números y símbolos. Caracteres en video intermitente. Caracteres gráficos.
Modos gráficos:	Texto alfanumérico - Gráficos de alta resolución/multicolor / combinación: texto/alta resolución o multicolor
Resolución:	320 x 200 puntos
Sonido:	2 generadores de tono o 1 de tono y otro de ruido blanco
Volumen:	9 niveles de volumen
Teclado:	66 teclas en total - 4 teclas de cursor - 4 teclas de función programadas (reprogramables a 8 funciones en total) Teclas de control de color - Tecla HELP fig. 2
Entradas salidas:	Ver fig. 1
Características:	BASIC 3.5 incorporando 100 comandos, incluyendo los específicos para gráficos y sonido. Monitor de código de máquina con 12 comandos - Capacidad de ventanas en pantalla

pantalla; el rectángulo puede rotar en cualquier ángulo y puede salir fuera de la línea.
Point: este comando le permite llenar cualquier área de la pantalla con cualquier color.
Draw, puede dibujar un punto, una línea entre dos puntos, y múltiple sección de puntos en el mismo comando.

BASIC 3.5

El equipo viene implementado para trabajar en lenguaje BASIC versión 3.5 pero posibilita trabajar en ASSEMBLER mediante el modo MONITOR. El BASIC 3.5 tiene 100 comandos, sentencias y funciones, comandos gráficos tales como circle, box, etc., funciones programas para manejo de disco como por ejemplo f3 (directorio), f5 (dsave), f2 (dload), etc., comandos especiales para sonido, y permite trabajar en lenguaje de máquina con doce comandos.

Listado de Comandos

AUTO	DELETE	HEADER	LOAD	RUN
BACKUP	DLOAD	HELP	NEW	SAVE
COLLEC	DSAVE	KEY	OPEN	SCNCLR
COPY	DIRECTORY	LIST	REN AME	SCRATCH

Listado de Instrucciones

BOX	DIM	GRAPHIC	NEXT	SCALE
CHAR	DRAW	GRAPHIC CLR	ON	SSHAPE
CIRCLE	END	CSHAPE	PAINT	STOP
CLEAR/HOOE	GET	IF THEN ELSE	POKE	SYS
CLOSE	GET	INPUT	PRINT	USING
CLR	GETKEY	INPUT	PRINT	TRAP
CMD	GOSUB	JOY	PUDEF	TROFF
COLOR	FOR... TO	STEP	READ	TRON
DATA	FLASH ON/OFF	LET	RESTORE	VOL
DEF FN	GOTO	MONITOR	RESUME	WAIT

Facilidades de programación.

Una de las características de esta computadora es la fácil comprensión y aprendizaje sobre el manejo del equipo. Existen numerosas ayudas tendientes a facilitar la tarea de programación. Por ejemplo:
el HELP, que permite detectar los errores de sintaxis;
el Auto-Joy, función útil para programas de juego que sirve para manejar el joystick,
las funciones DEC y HEX$ para efectuar conversiones entre valores hexadecimales y decimales.
Una particularidad del equipo es el "armado de ventanas" mediante la tecla ESC que posibilita encuadrar textos.

ASSEMBLER

Esta nota nos introduce en un lenguaje poderoso y rápido, más versátil que el Basic.



Existen controversias acerca de la calificación sobre este lenguaje. Hay programadores que sostienen que el lenguaje de máquina es de bajo nivel o primitivo debido a que requiere un profundo conocimiento del equipo, tal como localización de zonas de memoria, reserva de áreas y por ser un lenguaje propio a cada tipo de computadora en contraposición a lenguajes standard, definidos de alto nivel tales como BASIC, COBOL, FORTRAN

Diferencias entre Basic - Código de Máquina y Assembler

Ante todo cabe aclarar que se denomina lenguaje al medio o técnica para permitir la comunicación entre la computadora y el usuario.
Generalmente se confunde código de máquina con assembler.
La importancia de las diferencias nos obliga a detallarlas a fin de evitar errores conceptuales graves:

1º El BASIC es el lenguaje que permite en forma fácil y standard al usuario novato - en especial - programar un requerimiento. Es necesario, al ser ejecutado el programa un intérprete que transforme cada sentencia a código de máquina. Conclusión: este proceso demanda tiempo adicional.

2º En cambio, el código de máquina es

Figura 2 Proceso de ejecución de una órden "assembler"

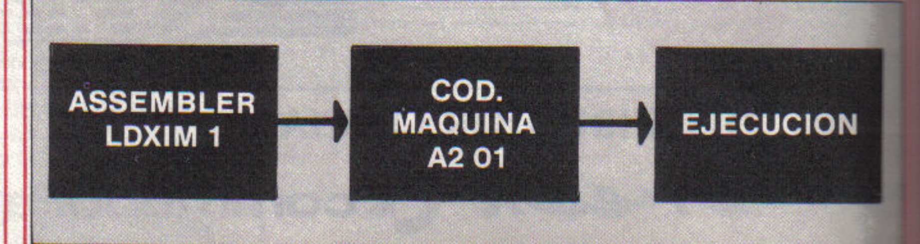
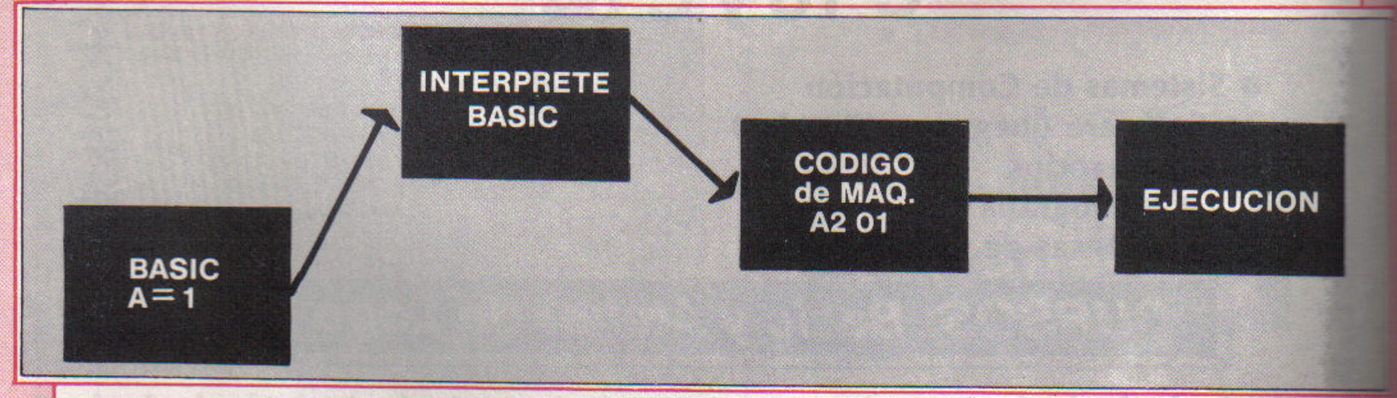


Figura 1 Proceso de decodificación de Basic a assembler



el lenguaje natural y propio del microprocesador 6510

3º Por último el assembler es el lenguaje que utiliza los códigos de máquina pero mediante el uso de nemotécnicos - facilitando, de este modo, la lectura y comprensión de los programas.

El assembler convierte símbolos (llamados nemónicos) de instrucciones de máquina y direcciones, en instrucción y direcciones reales, interpretables por el microprocesador.

Para facilitar la comprensión de lo expuesto, vamos a suponer un ejemplo sencillo: la suma de dos variables codificada en los tres lenguajes.

BASIC - El programa será:

```
10 A=1
20 B=2
30 C=A+B
40 PRINT C
```

Utilizando los códigos de máquina del 6510 será

```
A2 01
8E 84 03
A9 02
6D 84 03
8D85 03
60
```

La lectura posterior e interpretación de este listado es engorroso.

Codificando en assembler:

LDXIM	1	Carga el acumulador X con un 1
STX	900	Almacena el contenido del registro X en 900
LDAIM	2	Carga el acumulador con un 2
ADC	900	Adiciona a el acumulador el contenido de la localización de memoria 900
STA	901	Almacena el contenido del acumulador en la localización de memoria 901
RTS		Retorna desde la subrutina de código de máquina

Si se observa, la lectura del programa es mucho más fácil y ordenada.

Ventajas del "assembler"

El assembler es un lenguaje poderoso y rápido. Debemos considerar que al microprocesador 6510 le demanda un microsegundo completar un ciclo de trabajo.

Por lo tanto, todo lenguaje que se acerque a los códigos de máquina, en este caso el assembler, es más versátil y rápido comparado con el Basic. (Figura 1 y 2)

Para verificar la diferencia de velocidad entre Basic y el lenguaje de máquina, tipear los dos programas de fig. 3 y 4 respectivamente, los cuales efectúan el mismo proceso, consistente en limpiar la pantalla del modo de alta resolución. En basic demora alrededor de 30 segundos.

Necesidad del assembler

El assembler es necesario e indispensable para aquellos que deseen programar juegos, en los que se pretenda animación, rapidez en los movimiento y un preciso control de los jugadores.

También se lo suele utilizar en aplicaciones comerciales, tales como procesadores de palabra, manejo de base de datos, persiguiendo siempre por objetivo el mayor rendimiento del equipo.

Combinación de Basic y Lenguaje de máquina

Una de las dudas surgidas por el usuario es en qué casos es apropiado usar basic o assembler. El criterio a seguir es:

- en aquellos programas que incluyen fórmulas complejas, tal es caso de aplicaciones aritméticas, ingeniería, escogeremos BASIC.
  La razón por la cual no elegiremos trabajar en assembler es porque requiere técnicas de manipulación de los bits más compleja y difíciles que en Basic.
- es óptimo combinar ambos lenguajes. Podemos usar BASIC para rutinas donde la velocidad de ejecución no es vital, y el resto en lenguaje de máquina para lograr rapidez.

Mediante la utilización del comando SYS, llamamos a esas rutinas en lenguaje de máquina, en el momento de ser necesarias.

Consideraciones previas a la programación en assembler

1º Recordemos que el sistema operativo de la C-64 está escrito en lenguaje de máquina, existiendo numerosas rutinas incluídas en el área ROM, que podemos emplear en nuestros propios programas.

2º Hay que tener en cuenta que un programa en assembler permite implementar nuevos comandos, manipulables posteriormente por el BASIC

3º A todo programa escrito en assembler se le debe indicar el lugar en memoria (el comienzo) donde será almacenado.

4º El equipo dispone de una zona de memoria disponible para almacenar el programa. Generalmente, en programas cortos, se utiliza para guardar el programa, en el área del buffer del cassette, el cual se extiende desde el 828 al 1029, proveyendo en total 192 bytes.

Figura 3

```
10 I=49152: IFPEEK (49152) = 169 THENSYS 49152: END
20 READ A: LF A = 256 THEN SYS49152: END
30 POKE I,A: I = I + 1: GOTO 20
49152 DATA 169, 147, 32, 210, 255, 173, 24
49159 DATA 288, 9, 8, 141, 24, 208, 173
49166 DATA 17, 288, 8, 82, 141, 17, 208
49173 DATA 169, 0, 168, 133, 252, 169, 32
49180 DATA 133, 253, 169, 0, 145, 252, 200
49187 DATA 299, 249, 230, 253, 165, 253, 201
49194 DATA 64, 144, 241, 165, 197, 201, 60
49201 DATA 208, 250, 173, 17, 208, 41, 223
49208 DATA 141, 17, 208, 169, 21, 141, 24
49215 DATA 208, 96, 256
```

Tipiemos este programa en Lenguaje de máquina lo salvamos y en lugar de darle RUN, entramos SYS 49152, Comprobaremos la velocidad.

Figura 4

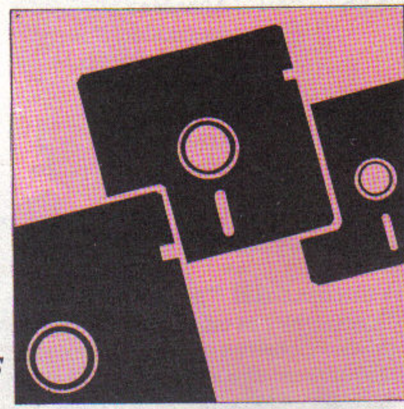
```
 5 PRINT "(CLR)": REM VERSION BASIC
10 POKE 53272, PEEK (53272) OR8
20 POKE 53265, PEEK (53265) OR32
30 FOR T = 8192 TO 8192+8000: POKE T,0: NEXT
40 GET AS: IF A$ = "" THEN 40: REM
   PRESIONE UNA TECLA
50 POKE 53265, PEEK (53265) AND 223
60 POKE 53272,21
```

Tipiemos este programa, salvémoslo en algún soporte y démosle RUN - será lento -

DISKER I

COMP.: COMMODORE
CONF.: BASICA
CLAS.: UTI
AUTOR: Sergio Freue

Soluciones para los problemas que se presenten cuando utilicemos la disketera Commodore y querramos ubicar un archivo.



El resfrío común y los directorios de Commodore son males cotidianos a los que no se les ha hallado solución aunque se trata de vivir con ellos.
Para el resfrío los remedios usados son muy conocidos. Pero, lamentablemente, no es así para los directorios, como sabrán todos los poseedores de una disketera Commodore, cuando luego de borrar varios archivos y poner otros nuevos sería imposible acertar en donde se ubicará el próximo archivo dentro del directorio, lo cual no sería grave si se usasen 5 ó 6 archivos por diskette (es decir es grave).

Pero más que esto, los problemas empiezan cuando los programas se van, más específicamente al usar el legendario comando '¡SO:...', con su maravillosa función para borrar más de un archivo, lo que más de una vez termina con uno o más archivos inocentes.
Después de esto es posible que se piense que es mejor sacrificar velocidad por confiabilidad y volver al cassette. Pero así como existen algunos remedios para el resfrío también es posible encontrar una solución para los directorios, y ahí es donde entra este programa.
El principio de funcionamiento de este programa es sencillo, realizándose en tres pasos principales:

1. Se leen todos los datos contenidos en el directorio (leyendo sector por sector) y guardándolos en el vector D$ (Dimensionado para contener hasta 144 nombres de archivos, el máximo permitido en un directorio de Commodore. Todos los días se aprende algo nuevo).
2. El usuario estructura el directorio a

100, 110	SHIFT - CLR, CRSR (abajo), CRSR (abajo)	**1010**	Los símbolos para CO$ son: CRSR (abajo, SHIFT-CRSR (abajo), f1, f3, f5, f7. Los símbolos para MS$ son: CLR, 23 veces CRSR (abajo), 6 veces CRSR (derecha)
128	SHIFT-CRSR (derecha)		
280	CLR, 22 veces CRSR (abajo)		
610, 630	SHIFT-CRSR (derecha)		
710	SHIFT-CLR		
750	CRSR (abajo)	**1090**	SP$ tiene 26 espacios

LINEA	DESCRIPCION
90-200	Inicialización y lectura del directorio en D$ ( ), usando una subrutina en lenguaje de máquina.
160	Si el archivo no es válido (no está cerrado o es del tipo "DEL") no incluirlo.
300-310	Preparación para mostrar en la pantalla los archivos
320-382	Listado en la pantalla de hasta 20 archivos a partir del número C1, hasta C3, indicándolos en PA (1-20)
390-410	Posicionar el cursor (sprite 0) y obtener la tecla presionada; saltar de acuerdo a ésta a la rutina correspondiente.
500	Cursor hacia abajo
520	Cursor hacia arriba
540	Establecer archivo sin proteger
550	Establecer archivo protegido
560-565	Borrar un archivo
570-680	Establecer separador
700-705	Asignar una posición a un archivo, si no es el último obtener otro.
710-740	Muestra de cómo será el nuevo directorio.
750-860	Escribir el directorio (Previa confirmación del usuario).
870	Fin del Programa
1000-1020	Dimensionar Matrices e inicializar variables (CO$ contiene los códigos de los comandos)
1040	Escribir (a partir de $C000) la subrutina en código de máquina para leer un sector
1060-1080	Inicializar los datos del sprite 0 (cursor)
1090-1100	Más inicialización de variables

su gusto; sirviendo las teclas de funciones para lo siguiente (una vez posicionado el cursor en el archivo deseado):

f1. El archivo indicado por el cursor será el próximo en el nuevo directorio.

f3. El archivo indicado será el próximo en el nuevo directorio, pero no podrá ser borrado mediante el comando 'SO:...'.

f5. El archivo indicado NO SERA INCLUIDO en el nuevo directorio, si el usuario lo confirma.

f7. En la próxima posición del directorio aparecerá un mensaje o separador (especificado por el usuario).

Para cada uno de los archivos del directorio original deberá usarse por lo menos una de estas opciones.

Cuando nos queden más archivos se pasará automáticamente a la tercera parte.

3. Los archivos, una vez estructurados a gusto del usuario, son escritos en el diskette, con lo que se tendrá un nuevo directorio.

La opción seleccionada (durante la

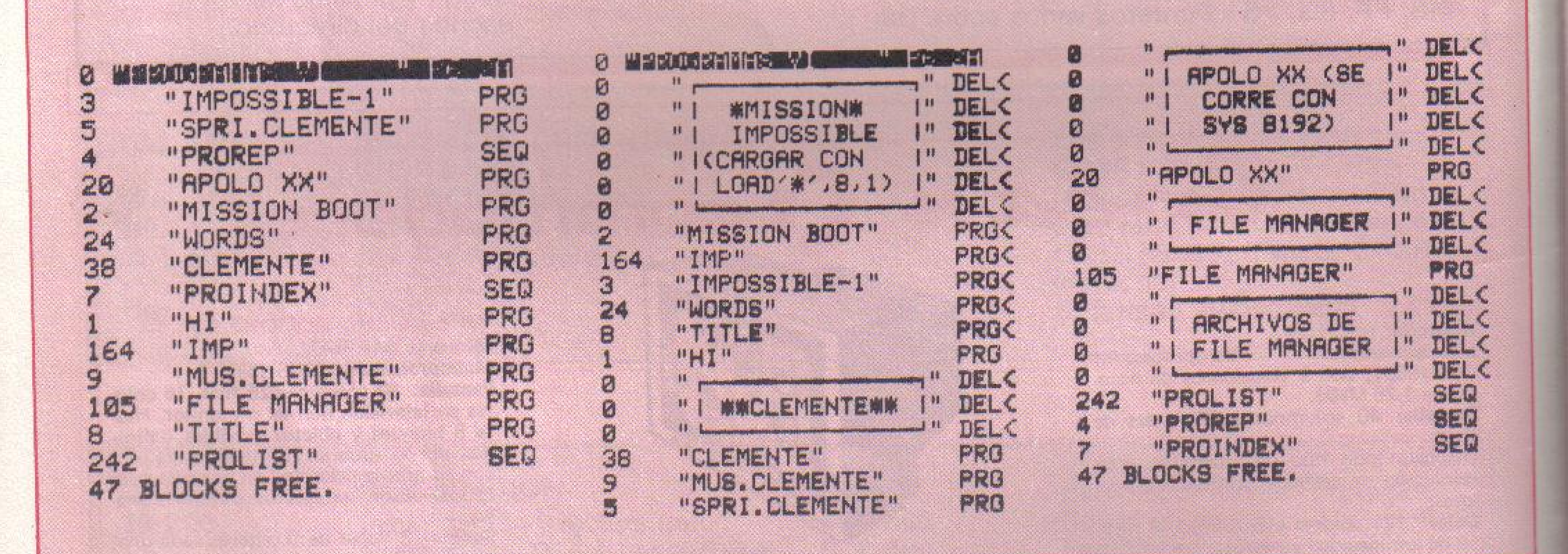
Descripción de las variables

VARIABLES	DESCRIPCION
PI,SE	Pista y sector
ER, ER$, PE, SE	Código de error, error, pista y sector
T, T1, A$, N$, N	Variables auxiliares
D$ (1-144)	Nombres de archivos en el directorio
D (0,1-144)	Indice sobre D$ ( ) de los archivos correspondiente de D$ ( ). (si es. 0 todavía no está ordenado; si es -1 el archivo está borrado)
UI	Indice al último archivo
MS, PP, PG, PS	Punteros varios sobre D$( )
TP$ (0-4)	Tipos de archivo (DEL, SEQ, PRG, USR, REL)
PA (1-20)	Indice sobre D$ ( ) del archivo en una línea determinada
DI (0-18)	Banderas de la disponibilidad de un sector (1-ocupado)
C	Posicion del cursor en la Pantalla
C1, C3	Indices al primer y último archivo sobre la pantalla
C2	Línea del último archivo sobre la pantalla
S1, S2	Sector y próximo sector a escribir del directorio.

segunda parte) por medio de la tecla f7, introducirá en el directorio un archivo de tipo DELeted, (Que, entre paréntesis, no es un tipo demasiado usado) que no podrá ser borrado, aunque no aparecerá si se corre otra vez este programa sobre el mismo diskette. Si durante esta opción se oprime la tecla '←', la opción sera abortada; y si se oprime 'RETURN' antes de tiper otra cosa, aparecerá en el directorio una línea.

Explicación de las "Técnicas novedosas"

LINEA	EXPLICACION
125	Se usa el comando wait para detectar cuándo ingresa un carácter en el buffer
400	Se busca el índice del comando (A$) dentro de CO$, y se salta a la rutina adecuada (dada por la posición en CO$)
500, 520, etc.	Usando el resultado de una comparación, (que puede ser 0 o -1) se salta, con el comando ON...GOTO a una línea si la condición se cumple, o alternativamente a otra si no se cumple (IF...THEN...ELSE)
550	Para proteger un archivo se prende el sexto bit del byte correspondiente al tipo de archivo
590	Se salta a distintas líneas si la tecla oprimida fue alguna de las indicadas (DEL, RETURN o " ")
600	Se chequea que la tecla oprimida no corresponda a un caracter de control (cambio de color, movimiento de cursor, etc) observandd o si los bits 5 y 6 están apagados.



```
0    [illegible]
3     "IMPOSSIBLE-1"      PRG
5     "SPRI.CLEMENTE"     PRG
4     "PROREP"            SEQ
20    "APOLO XX"          PRG
2     "MISSION BOOT"      PRG
24    "WORDS"             PRG
38    "CLEMENTE"          PRG
7     "PROINDEX"          SEQ
1     "HI"                PRG
164   "IMP"               PRG
9     "MUS.CLEMENTE"      PRG
105   "FILE MANAGER"      PRG
8     "TITLE"             PRG
242   "PROLIST"           SEQ
47 BLOCKS FREE.
```

```
0    [illegible]
0    "┌────────────────┐"  DEL<
0    "|   *MISSION*    |"  DEL<
0    "|   IMPOSSIBLE   |"  DEL<
0    "|(CARGAR CON     |"  DEL<
0    "| LOAD'*',8,1)   |"  DEL<
0    "└────────────────┘"  DEL<
2    "MISSION BOOT"        PRG<
164  "IMP"                 PRG<
3    "IMPOSSIBLE-1"        PRG<
24   "WORDS"               PRG<
8    "TITLE"               PRG<
1    "HI"                  PRG
0    "┌────────────────┐"  DEL<
0    "| **CLEMENTE**   |"  DEL<
0    "└────────────────┘"  DEL<
38   "CLEMENTE"            PRG
9    "MUS.CLEMENTE"        PRG
5    "SPRI.CLEMENTE"       PRG
```

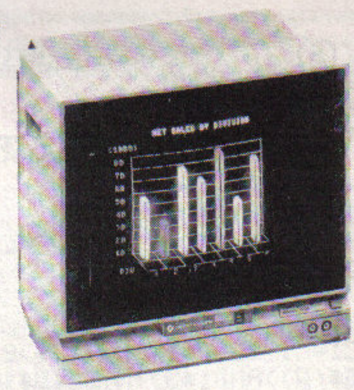
```
0    "┌────────────────┐"  DEL<
0    "| APOLO XX (SE   |"  DEL<
0    "|  CORRE CON     |"  DEL<
0    "|  SYS 8192)     |"  DEL<
0    "└────────────────┘"  DEL<
20   "APOLO XX"            PRG
0    "┌────────────────┐"  DEL<
0    "| FILE MANAGER   |"  DEL<
0    "└────────────────┘"  DEL<
105  "FILE MANAGER"        PRG
0    "┌────────────────┐"  DEL<
0    "| ARCHIVOS DE    |"  DEL<
0    "| FILE MANAGER   |"  DEL<
0    "└────────────────┘"  DEL<
242  "PROLIST"             SEQ
4    "PROREP"              SEQ
7    "PROINDEX"            SEQ
47 BLOCKS FREE.
```

```
90 GOSUB1000
100 PRINT"[illegible]>>PONER EL DISKETTE, OPRIMIR UNA TECLA.":WAIT198,1:GETA$
110 OPEN15,8,15:OPEN2,8,2,"#":PRINT"[illegible] EL DIRECTORIO USA:"
120 PI=18:SE=1:UL=1
125 IFPI=0THENPRINT" OPRIMA UNA TECLA.":WAIT198,1:GETA:GOTO300
128 PRINT"SECTOR:"PI"[illegible],"SE
130 PRINT#15,"U1:2 0";PI;SE:INPUT#15,ER,ER$,PE,SE:IFERTHENCLOSE2:CLOSE15:END
140 SYS49152:PI=PEEK(52992):SE=PEEK(52993)
150 FORI=52992TO53216STEP32
160 N$="":IF(PEEK(I+2)AND128)=0ORPEEK(I+2)=192THEN190
170 FORI1=2TO31:N$=N$+CHR$(PEEK(I+I1)):NEXTI1
180 D$(UL)=N$:UL=UL+1
190 NEXTI
200 GOTO125
300 CLOSE2:CLOSE15:UL=UL-1:IFUL=0THENPRINT"NO HAY NINGUN ARCHIVO VALIDO.":END
305 PP=1:MS=UL+1:PG=UL
310 C1=1:C=1
320 C2=1:C3=C1:POKE53269,0
```

```
330 PRINT"♥     PROGRAMA  ","TIPO"
335 PRINT"  ──────────  ","────"
340 IFD(1,C3)THEN360
350 PRINT"    "MID$(D$(C3),4,16),TP$(ASC(LEFT$(D$(C3),1))AND7):PA(C2)=C3:C2=C2+1
360 C3=C3+1:IF(C3<=UL)AND(C2<21)THEN340
370 C3=C3-1:C2=C2-1:POKE53269,1:IFC2=0THEN310
380 PRINT"▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒   ┌──────────────────────────────────────────┐"
381 PRINT"       │                                      │"
382 PRINT"       └──────────────────────────────────────┘";
390 GETA$:POKE53249,59+8*C:IFA$=""THEN390
400 CO=0:FORI=1TOLEN(CO$):IFA$=MID$(CO$,I,1)THENCO=I
410 NEXT:ONCOGOTO500,520,540,550,560,570
420 GOTO390
500 C=C+1:ON2+(C<=C2)GOTO390:C1=C3+1+UL*(C3=UL):C=1:GOTO320
520 C=C-1:ON2+(C>0)GOTO390:C1=C1-20:C=1:ON2+(C1>0)GOTO320,310
540 N=PA(C):D$(N)=CHR$(ASC(D$(N))AND143)+MID$(D$(N),2):GOTO700
550 N=PA(C):D$(N)=CHR$(ASC(D$(N))OR192)+MID$(D$(N),2):GOTO700
560 PRINTMS$"BORRARLO ?":FORI=1TO10:NEXT:PRINTMS$"            ":GETA$
563 FORI=1TO30:NEXT:IFA$=""THEN560
565 ON2+(A$<>"S")GOTO390:D(1,PA(C))=-1:PG=PG-1:C=1:ON2+(PG=0)GOTO710,320
570 ON2+(MS>144)GOTO390:N$=LEFT$(SP$,16):N=1:PRINTMS$"SEPARADOR:";
590 GETA$:ON-(A$=CHR$(13))-2*(A$=CHR$(20))-3*(A$="←")GOTO650,630,680
600 IF(A$="")OR((ASC(A$+Z$)AND96)=0)THEN590
610 N$=LEFT$(N$,N-1)+A$+MID$(N$,N+1):N=N+1:PRINTA$;:IFN=17THENN=16:PRINT"▌";
620 GOTO590
630 ON2+(N=1)GOTO590:PRINT"▌ ▌";:N=N-1:GOTO590
650 D$(MS)=TT$+N$+Z1$:D(0,PP)=MS:MS=MS+1:PP=PP+1
660 IFN=1THEND$(MS-1)=TT$+"----------------"+Z1$
680 FORI=1TO400:NEXT:PRINTMS$;SP$;:GOTO390
700 D(0,PP)=PA(C):D(1,PA(C))=PP:PG=PG-1:PP=PP+1:IFC=C2THENC=1
705 IFPGTHEN320
710 POKE53269,0:POKE198,0:PP=PP-1:PRINT"♥EL NUEVO DIRECTORIO SERA ASI:"
720 FORI=1TOPP:N$=D$(D(0,I)):N=ASC(N$):L=ASC(MID$(N$,29))+256*ASC(MID$(N$,30))
730 PRINTMID$(STR$(L)+"       ",2,5)MID$(N$,4,16)" "TP$(NAND7);
732 IFNAND64THENPRINT"<":GOTO740
735 PRINT
740 FORI1=1TO200:NEXT:NEXT
750 PRINT"▒▒MODIFICAR EL DIRECTORIO DEL DISKETTE ?":WAIT198,1:GETA$
760 IFA$<>"S"THENFORI=1TO144:D(0,I)=0:D(1,I)=0:NEXT:GOTO305
770 S1=1:FORI=0TO18:DI(I)=0:NEXT:DI(0)=1:DI(1)=1:C=1:OPEN15,8,15:OPEN2,8,2,"#"
780 PS=1:S1=1:S2=4
790 FORI=0TO7:IFITHENN1$=Z$+Z$:GOTO810
800 N1$=CHR$(18)+CHR$(S2):IFPP-PS<8THENN1$=Z$+CHR$(255)
810 IFPS>PPTHEND$(D(0,PS))=Z1$+Z1$+LEFT$(Z1$,8)
820 PRINT#15,"B-P:";2;32*I:PRINT#2,N1$;D$(D(0,PS));:PS=PS+1:NEXTI
830 PRINT#15,"U2:2 0 18";S1:S1=S2
840 S2=S1+3:IFS2>18THENS2=S2-18:PRINT#15,"B-P:2 255"
850 IFDI(S2)THENS2=S2+1:ON2+(S2<19)GOTO850:S2=S2-18:GOTO850
860 IFPS<=PPTHEN790
870 CLOSE2:CLOSE15:END
999 END
1000 DIMD$(144),D(1,144),PA(20),DI(18)
1010 Z$=CHR$(0):CO$="▒▒▒▒":MS$="▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▌▌▌▌▌▌":Z$=CHR$(0)
1020 FORI=0TO4:READTP$(I):NEXT
1030 DATA DEL,SEQ,PRG,USR,REL
1040 FORI=0TO18:READN:POKE49152+I,N:NEXT
1050 DATA162,2,32,198,255,160,0,32,228,255,153,0,207,200,208,247,76,204,255
1060 FORI=0TO14:READN:POKE832+I,N:NEXT
1069 FORI=15TO63:POKE832+I,0:NEXT
1070 DATA168,6,0,84,3,128,127,255
1075 DATA240,84,3,128,168,6,0
1080 POKE2040,13:POKE53287,0:POKE53264,0:POKE53248,26
1090 Z1$="":FORI=1TO11:Z1$=Z1$+Z$:NEXT:SP$="                "
1100 TT$=CHR$(192)+CHR$(18)+CHR$(1)
1110 RETURN
```

REGISTRO DE CONTROL DE MEMORIA

Registro del chip Vic-II (localización 53272) que afecta virtualmente a todas las operaciones gráficas



Función

Este registro afecta virtualmente a todas las operaciones gráficas.
Determina la dirección base de dos áreas muy importantes:
1) - la matriz de video
2) - área de datos punteados de caracteres
Podemos utilizar este registro, para mover el comienzo de memoria de pantalla a cualquier lugar, delimitado por el espacio de 1 K, dentro de los 16 K del área de memoria de VIC-II.
Valores del registro de control de memoria.
Esste registro puede tener hasta 8 bits, cada valor representa lo siguiente.
BIT 0: No tiene uso
BITS 1 - 3: dirección base de datos punteados de caracteres de texto dentro del espacio de la dirección VIC-II
BITS 4 - 7: dirección base de matriz de video dentro del espacio de la dirección VIC-II
BITS 1 - 3

Definición

Estos bits son usados para fijar la ubicación del área de datos punteados de caracteres. Esta área es donde se almacenan los datos que definen la forma de los caracteres impresos en pantalla.
Los bits 1 a 3 pueden representar cualquier número par de 0 a 14.
Ese número a su vez representa el offset para de 1 K del área de datos de caracteres, desde el principio de la memoria de VIC-II.
Por ejemplo: si todos estos bits son fijados a O, significa que la memoria de caracteres ocupa los primeros 2 K de la memoria de VI-II. Si son iguales a 2, el área de datos comienza en 2 x 1 K o sea 2 x 1024 que es igual a 2048 bytes del principio de la memoria VIC (Ver tabla figura 1)
El valor de omisión de este NIBBLE es 4. Esto fija la dirección del área de datos punteados de caracteres en 4096 ($1000) que es la dirección inicial de donde el chip VIC-II ubica al ROM de caracteres.

Manejo de caracteres

El juego normal de caracteres, conteniendo las mayúsculas y gráficos, ocupan los primeros 2 K de ese ROM. En cambio, el juego alternativo de caracteres (mayúsculas y minúsculas) usan los segundos 2 K. Por lo expresado, para cambiar el juego alternativo de caracteres, se debe cambiar el valor de este NIBBLE a 6 mediante:
POKE 53272, PEEK (53272) AND 253

Modo BITMAP

Bajo este modo, el nibble inferior controla la ubicación de los datos de pantalla de bitmap.
Como esta área de datos puede comenzar sólo a un offset de 0 a 8 K del principio de memoria del VIC-II, solo el bit 3, del registro de control de memoria, significa algo en modo bitmap.
Si bit 3 tiene un 0, el offsset es 0 y si tiene un 1, el offset es 8192 (8 K).
BITS 4 - 7

Definición

Este nibble indica la dirección inicial del área de matriz de video.
Esta área de matriz de video es la correspondiente a las localizaciones de pantalla con una capacidad de 1024 bytes y que contiene los códigos que representan los caracteres de texto que se vuelcan a la pantalla.
Además de esto, los últimos 8 bytes de esta área se usan como señaladores que designan cuál bloque de 64 bytes de memoria VIC-II serán usados como datos de forma para cada sprite.
Estos 4 bits pueden representar números de 0 al 15. Estos números significan el offset (incrementos de 1 K) desde el comienzo de la memoria VIC-II de matriz de video. Por ejemplo: el patrón de bit de omisión es 0001. Esto indica que la matriz de video está desplazada en 1 K con respecto al comienzo de la memoria VIC-II, que es el lugar normal para el inicio de la memoria de la pantalla.

Debe recordarse que el valor de bit de este número es 16 veces lo que indique el patrón de bit, porque nos estamos refiriendo a los bits 4 a 7. Por lo tanto el 0001 en el nibble superior tiene un valor de 16.
Mover la pantalla
De acuerdo a lo expuesto, podemos mover el comienzo de la dirección de memoria de la pantalla, teniendo en cuenta el espacio de 1 K, dentro de los 16 k disponibles.
Ahora bien, no es suficiente con cambiar el registro si se desea usar el editor de línea BASIC. El editor se referencia en ubicación 648 ($288) para determinar dónde imprimir caracteres de pantalla.
Si sólo se cambia la ubicación de la matriz de video sin cambiar el valor en 648, el BASIC continuará imprimiendo caracteres en el área de memoria comenzando en 1024, aunque esa área ya no se encuentre volcada en pantalla. El resultado será no poder visualizar lo que se está tipeando en el teclado.
Para solucionarlo se debe indicar "POKE 648" con el número de página de la dirección inicial de memoria de pantalla. Ejemplo: Número de página- dirección / 256
Recuerde: la dirección inicial real de memoria de pantalla, depende no sólo del offset desde el comienzo de memoria VIC-II en el registro, sino también de cuál banco de 16 K está siendo usado como memoria VIC-II.
Ejemplo: Si el área de pantalla comienza a 1024 bytes del principio de memoria VIC-II y el chip de video está usando banco 2 (32768) - 49151) la dirección inicial real de la memoria de pantalla es:
32768 1024 33792 ($ 8400)

NUMERO	DECIMAL
0	0
2	2048
4	4096
6	6144
8	8192
10	10240
12	12288
14	14336

INGRESO IRRESTRICTO A LA COMPUTACIÓN.

A partir de hoy, Commodore entra a todas las aulas. Con Proceda.

Ahora, la computación es la mejor escuela.
Con Proceda. Y su exclusiva distribución del computador Commodore en el área educativa.
Sumando a estos avanzados equipos, el más desarrollado software y un constante servicio de mantenimiento y actualización.
Con esta designación, que nos enorgullece por su gran trascendencia, todos los estudiantes podrán ingresar al futuro.
Haciéndolo realidad hoy. Con Drean Commodore y Proceda.

Centro Especializado en Computación Personal:
Av. Córdoba 650 (Casi Florida) - Tel. 392-7611/8478
Casa Central: Av. Pueyrredón 1770. Tel. 821-2051

Drean
Commodore

Sucursal Córdoba: Boulevard Reconquista 178. Tel. 36-207 y 39-520.
Centro: Peatonal San Martín 149 (Córdoba). Tel. 24-447.



Informática Integral

por Diana Decunto

INFORME ESPECIAL

Para aquellos que hace muy poco consiguieron una Commodore 64, les ofrecemos este informe, que deben leer con atención para no hacer desastres (no los exime de estudiar el manual).

Los pasos previos al uso del computador deberán ser conocer la configuración y las cualidaddes del equipo.
Para lograr estos objetivos aconsejamos leer detenidamente el manual o guía que viene con el equipo y material bibliográfico complementario.
Es fundamental conocer - paso posterior a la lectura - lo relacionado a la conexión del hardware a fin de evitar el uso indebido o perjudicial del equipo.
Algunas recomendaciones útiles al usuario novato al efectuar las conexiones son:
1º Verificar la fuente de alimentación. Esta puede estar a 110 V y recorde - mos que en dicho caso necesitaremos un transformador que adapte la corriente de 110 a 220 V.
2º Las conexiones estarán perfectamente realizadas si al encender el equipo, la luz de poder (power) se encuentra de color rojo y se proyecta la imagen en el televisor o monitor, tal como se muestra en figura 1.

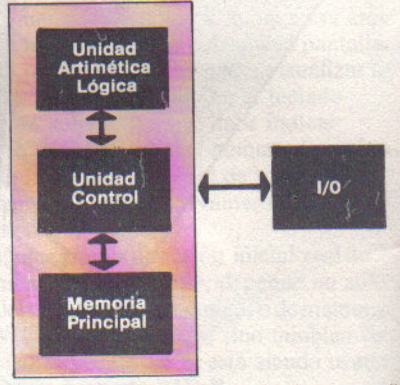
Figura 1

```
***COMMODORE 64 BASIC V2 *** (c)
64 K RAM SYSTEM 38911 BASIC BYTES FREE (d)

READY
(b) ■
                    (a)
```

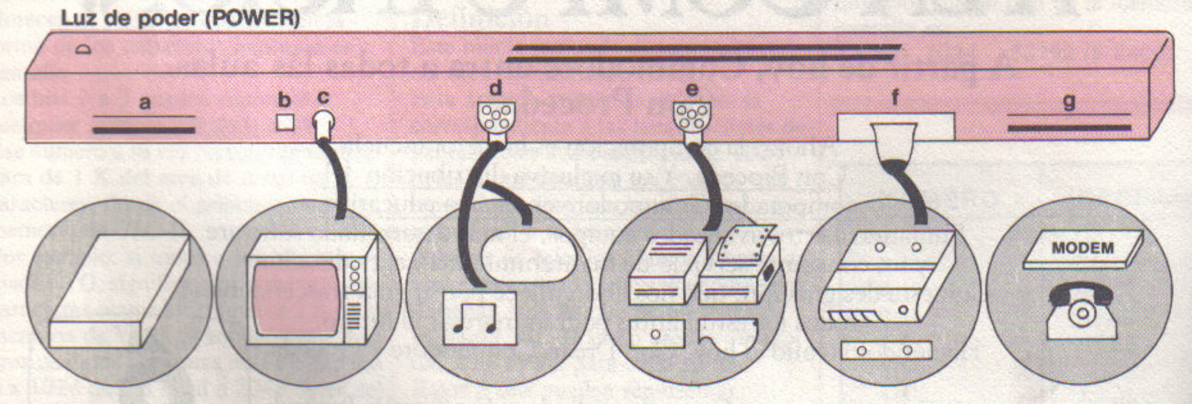
(a) Fondo de pantalla (localización 53281)
(b) Borde (localización 53280)
(c) Marca y Basic Versión 2.0
(c) Memoria total 64 k - 38 K para programación.

Figura 3

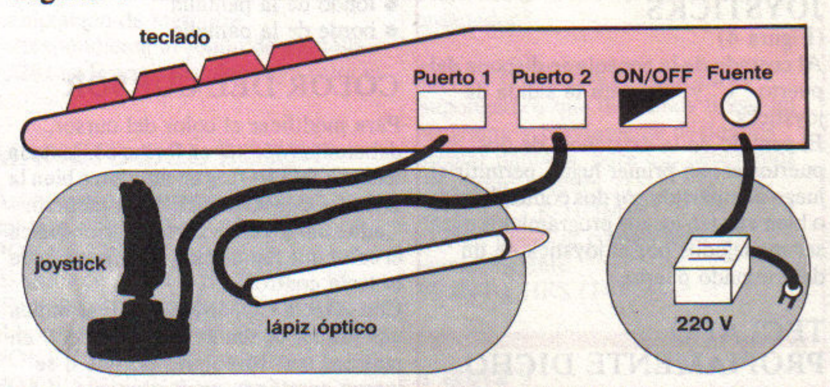


La función de la CPU, podemos sintetizar, es aquella que controla las operaciones de E/S y cálculos aritméticos.
(Además de la transferencia de datos entre memoria y registros)

Figura 2



a) cartucho **b)** modulador - **c)** salida a TV **d)** salida AUDIO/VIDEO (amplificador de música **e)** salida serial - **f)** salida a cassettera - **g)** modem monitor telefónico



3º Debemos verificar periódicamente que la fuente de alimentación no recaliente, ni tampoco el equipo o los periféricos.

ESTUDIO DEL TECLADO

(Ver figura 2)

Es importante antes de comenzar a programar o utilizar la computadora conocer cada una de las partes que la componen y también la función que cumplen.

Cabe destacar que, a nivel de Hardware, la consola trae en conjunto:

1) CPU o Unidad Central de Proceso
2) Teclado
3) salida serial para periféricos
4) salida a cassettera o interface
6) modulador y salida para TV
7) salida para cartridge o cartucho
8) salida a MODEM
9) 2 puertos para Joysticks, paddles o lápiz óptico.

CPU (Unidad Central de Proceso)

La consola, en su estructura, posee internamente un conjunto de chips o pastillas de silicio, cada una encargada de cumplir una función determinada. Las funciones de la Unidad Central de Proceso están agregadas en una única pastilla de silicio y recibe el nombre de MICROPROCESADOR. Existen diferentes tipos de microprocesadores. En el caso de la Commodore 64, es el 6510 compatible con el 6502. (Figura 3)

SALIDA a DRIVE o IMPRESORA

El equipo Commodore trabaja con salida serial, permitiendo conectar un drive o impresora.

SALIDA TV ó SALIDA MONITOR (AUDIO VIDEO)

Si utilizamos la salida TV, la imágen y sonido será transmitida a través del televisor.

También se puede conectar a un monitor y equipo de alta fidelidad.

En todos los casos la pantalla disponible será de 40 columnas por 25 líneas.

SALIDA A MODEM TELEFONICO

El equipo preve una salida para conexión del modem telefónico. La función del modem es transmitir o recibir información a partir de otro usuario que utilice un equipo con dicha conexión.

SALIDA PARA CARTRIDGE

En la configuración está prevista la entrada de lo que se denomina CARTRIDGE, CARTUCHO o TARJETA.
El cartridge es una tarjeta de memoria ROM externa preprogramada. Numerosos juegos o programas utilitarios vienen en este dispositivo.

VENTAJAS DEL CARTRIDGE

La utilización de este dispositivo evita la carga de los programas, debido a que en el momento de ser insertado, el programa ya reside automáticamente en la memoria del equipo. En consecuencia evita la utilización de periféricos (cintas odiskettes) para la carga de programas, pero no permite el salvado de nuestros propios programas.

SALIDAS PARA JOYSTICKS

(Figura 4)
Al costado de la consola se dispone del puerto "1" y "2" para la salida de joysticks.
El sentido de la existencia de dos puertos es, en primer lugar, permitir el juego compartido por dos contendientes o bien existen juegos programados para ser manejados por el joystick de un determinado puerto.

TECLADO PROPIAMENTE DICHO

El teclado del equipo Commodore, muy dúctil y de suave manejo, podemos dividirlo para facilitar el estudio en:

- Teclado numérico (parte superior)
- Teclado alfabético
- Teclado de control
- Tecla Commodore
- Caracteres en modo reverso
- Caracteres en modo gráfico (mayúscula y minúscula)
- 2 teclas de control de cursor
- RUN/STOP - RESTORE
- RETURN
- 4 teclas de función programables a 8

MANEJO DEL COLOR

Una de las ventajas de este equipo es la capacidad gráfica y los colores. Podemos puede manejarnos con 16 colores. Trabajando en modo normal es posible modificar el color del:

- cursor
- fondo de la pantalla
- borde de la pantalla

COLOR DEL CURSOR

Para modificar el color del cursor, deberemos apretar en forma conjunta la tecla CONTROL y un número o bien la tecla COMMODORE y un número. Cada número en la parte inferior indica el color que representa, si es presionada la tecla control.
Con la tecla COMMODORE se indica una grama de dicho color. Es decir en realidad son 16 colores porque 8 se logran apretando conjuntamente control y un número, y los otros 8 se consiguen mediante la presión de la tecla COMMODORE y un número (Figura5)
Otra opción —no tan fácil— es recurrir al código ascii. (por ejemplo, si deseamos el cursor de color blanco, encendido el equipo, tipear:
print chr$ (05) < return >

MANEJO DEL FONDO DE LA PANTALLA

Si observamos detenidamente veremos que la pantalla posee un fondo y un recuadro. Dicho fondo consta en modo normal de 40 líneas por 25 filas y en alta resolución es de 320 x 200.
Nosotros podemos modificar el color del fondo. Para ello debemos tipear:
POKE 53281, n <return>

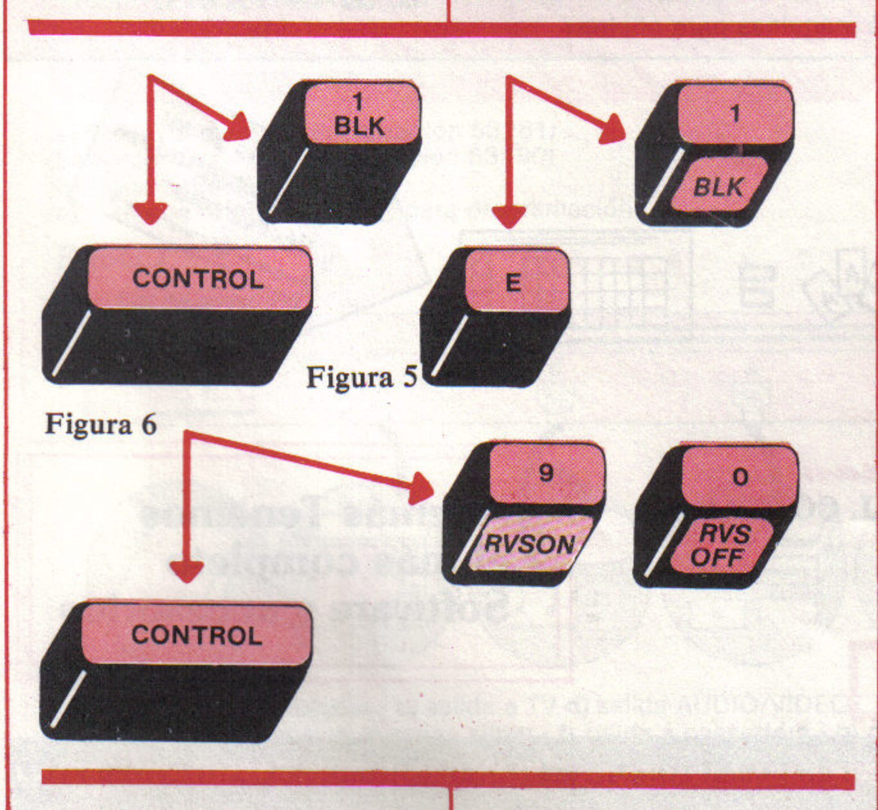


Figura 5

Figura 6

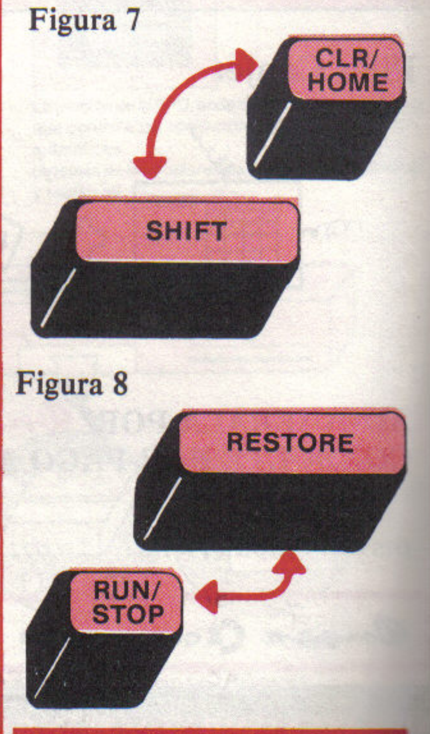


Figura 7

Figura 8

El valor de n será n ≥ 0 y n ≤ 15
Tipear este comando significa que la localización de memoria correspondiente al fondo de pantalla 53281 se le envie el color cuyo valor es n.

MANEJO DEL BORDE

Asimismo podemos modificar el color del borde tipeando:
POKE 53280, n return
El valor de n será n 0 y n 15
Ejemplo: si deseamos poner el fondo y el borde de color negro debemos tipear
POKE 53281, 0 <return>
POKE 53280,0 <return>
Cursor color blanco tipear: Control 2

Control 9 y Control 0

Si apretamos Control 9 todos los caracteres se editarán en modo reverso (ON). Caso contrario con control 0 el modo reverso queda cancelado (OFF) (Figura 6)

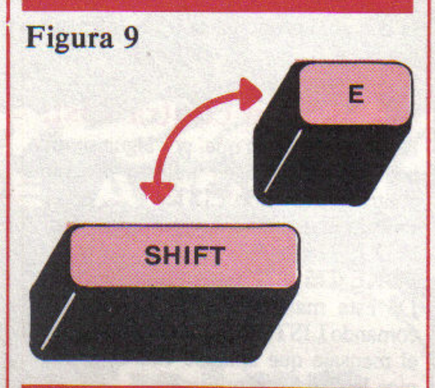
RETURN

La importancia de esta tecla consiste en indicar al computador la ejecución de un determinado proceso. Es el medio de comunicación entre el usuario y el ordenador.

LIMPIAR PANTALLA

Supongamos que deseamos tipear un programa y previamente decidimos limpiar pantalla, para eso deberemos presionar: (Figura 7)
SHIFT/CLR-HOME
O bien tipear el código ascii correpondiente
PRINT CHR$ (147)

Figura 9



INST/DEL

Es una tecla que sirve cuando escribimos programas para insertar o deletrear caracteres.

RUN STOP / RESTORE

Estas teclas, conjuntamente, sirven para interrumpir la ejecución de un programa. En ciertas aplicaciones estas teclas no se pueden utilizar, debido a que por software podemos desactivar su uso a fin de lograr protecciones. (figura 8)

MAYUSCULA Y MINUSCULA

(Figura 9)

Para trabajar con minúsculas Ud. debe presionar simultáneamente
SHIFT-COMMODORE
Dejándolo en modo ON. Para regresar al modo de Mayúsculas deberá repetir la operación.
El manejo de las teclas de cursor, las de función (f1 a f8). Y caracteres gráficos, por su extensión y tratamiento especial lo desarrollaremos en futuras publicaciones.

Joystick

En algunos programas donde se requiere JOYSTICK debemos en algún momento presionar una tecla para continuar. ¿Por qué no usar el botón del JOYSTICK en reemplazo de la tecla?
Con las siguientes líneas podemos parar el programa hasta que dicho botón sea presionado.
WAIT 145,16,16 para el JOYSTICK # 1
WAIT 56464,16,16 para el JOYSTICK # 2

Datassette

Si el televisor o el monitor tiene capacidad de sonido podemos usarlo para oir los programas en el datassette.
De tal manera, si tenemos algún problema de grabación podremos detectarlo con más facilidad.
Para hacer posible esto tipeamos:
POKE 54296,V donde V debe ser un valor de 0 (mínimo) a 15 (máximo) y luego sube el volumen de tu TV hasta el máximo.
Cuando grabemos, carguemos o verifiquemos, podremos escucharlo fuerte y claro.

Laberinto

Si tipeamos esta línea la computadora dibujará un laberinto eterno, siempre de forma distinta.
8 PRINT CHR$ (205+RND (8)): GOTO 8
Para lograr un laberinto en colores digitamos:
8 PRINT CHR$ (205.5 + RND (8)) CHR$ (149+ RND (8) *11);:GOTO 8

Rundy

¿Tipearon algun vez RUN sobre un READY?. Si lo hicieron la computadora probablemente ha indicado el conocido mensaje Undef'd Statement error (error de informe indefinido)9
Pero si el programa comienza en la línea cero podrá ser ejecutado sin importar si quedan caracteres en la pantalla luego del RUN.

En cualquier lugar

POKE 214,F: PRINT: POKE 211, C: PRINT "TEXTO"
Con estas instrucciones imprimiremos en cualquier lugar de la pantalla, donde F es la FILA y C es la COLUMNA.
Si se ejecuta este programa se comprobará que la impresión puede iniciarse/pararse con solo presionar la tecla antes nombrada.

Protección contra List:

Hay ocasiones en que, por algún motivo, necesitamos proteger nuestro programa contra el comando LIST.
Esto lo podemos conseguir tipeando:
POKE 774, 148
POKE 775, 227
De esta manera cuando ingresemos el comando LIST, se imprimirá en la pantalla el mensaje que aparece cuando prendemos la C-64 por primera vez.
Además, todo programa almacenado en la memoria es borrado.
Otra manera de proteger el programa es haciendo:
POKE 774,0
Aquí solo se imprimirán los números de línea de nuestro programa cuando el comando LIST sea ejecutado.
Para regresar al modo normal se debe tipear:
POKE 774, 26

Repetición automática

Como habremos podido comprobar, el cursor y la barra espaciadora son las únicas teclas que tienen repetición automática.
Con la siguiente instrucción todas las teclas de computador tendrán repetición automática.
POKE 650, 128
Si queremos que ninguna tecla tenga repetición automática tipeamos la siguiente instrucción:
POKE 650,64

Avisador de fin de proceso

Esta rutina genera un leve "BEEP". La podemos utilizar para que la C-64 nos avise cuando haya culminado una determinada tarea.

```
10 REM PGM PRINCIPAL
600 REM FIN PGM
610 REM COMIENZO RUTINA BEEP
620 POKE 54273,80: POKE 54278,249: POKE 54296,15
630 POKE 54276,07: POKE 54276,16
640 REM FIN RUTINA
650 END.
```

La frecuencia de este "BEEP" (tono) puede ser modificada en el primer POKE. Si incrementamos el contenido de la dirección (decimal) 54273 aumentaremos la frecuencia. Todo lo contrario ocúrre si disminuimos el valor (recordar que éste debe estar comprendido entre 0 y 255).

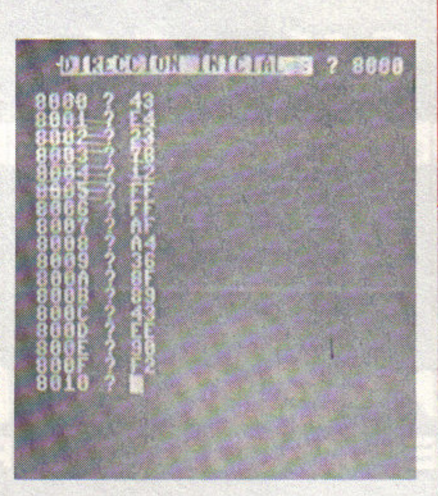
Suspensión temporal de ejecución

Esta rutina suspende la ejecución del programa actual hasta que la tecla COMMODORE es la que está ubicada a la derecha de la tecla SHIFT) sea presionada. Si se deja de oprimir, la ejecución del programa vuelve a suspenderse.
Podemos usar esta rutina cuando debemos controlar o sincronizar procesos, proveniente del exterior o, simplemente, para desarrollos de juegos:
EJEMPLO:

```
10 FOR I = 1 TO 100
20 PRINT I,
30 REM COMIENZO RUTINA
40 WAIT 654, 10
50 REM FIN RUTINA
60 NEXT I
```

"ASEMBLER 1.1"

TIPO : UTILITARIO
MEM : 4.5 Kb
COMP : COMMODORE 64
AUTOR: CRISTIAN JAVIER PARODI



Este software hace que la C-64 se convierta en un Kit de microprocesador, logrando que el programador escriba sus programas directamente en absoluto, sin hacer molestas cuentas

Descripción

Comúnmente cuando se desea escribir un programa en el lenguaje máquina de la C-64 (y en general para la mayoría de las Homes Computers) se debe, una vez codificado el programa, pasar las instrucciones de hexadecimal a decimal, ya sea a mano o a través de un programa, y realizar la carga en memoria utilizando la sentencia POKE juntamente con los DATA donde se encuentran los datos e instrucciones en decimal. Esta es una de las tantas formas de introducir un programa en lenguaje máquina.

Generalmente este procedimiento se utiliza cuando el programa funciona correctamente. Es decir que una vez cargado, sólo debemos hacer SYS sin correr el riesgo de que la computadora se "cuelgue".

Así como este método tiene sus ventajas (velocidad de carga, requerimientos de memoria mínimos para su implementación), también tiene sus desventajas. Una de ellas es la imposibilidad de sacar un listado que nos represente en forma directa las instrucciones y los datos cargados. Con esto queremos decir que sería más útil encontrarnos con "FF" en lugar de "255", más aún si debemos detectar un salto erróneo o determinar el valor del contador de programa (PC) en un instante dado. Otra de las desventajas, y creo que la más importante, es la de no poder experimentar en forma completa con el asembler del microprocesador 6510.

El programa "ASEMBLER 1.1" hace que la C-64 se convierta en un Kit de microprocesador, logrando que el programador escriba sus programas directamente en absoluto, sin tener que hacer molestas cuentas u otras funciones acordes a la carga decimal. Además, le suministra un total de dieciocho (18) comandos divididos en catorce (14) para depuración, ejecución, modificación, entrada/salida y cuatro (4) para uso del Disk Drive. Está escrito en lenguaje BASIC y ocupa 4.5 Kb de memoria. El programador dispone de, aproximadamente, 33 Kb libres para escribir sus programas.

Su funcionamiento se divide en dos modos:

i) Modo carga
ii) Modo comando

Funcionamiento

i) Modo Carga:

Este modo, como su nombre lo indica, permite la carga del programa instrucción por instrucción. La dirección de memoria inicial del mismo estará comprendida entre las direcciones hexadecimales 1B00 y 9FFF (comienzo y fin de la memoria libre). Esto debe recordarse ya que lo primero que efectuará el editor (así definimos el programa) es preguntar por la dirección inicial a través del siguiente mensaje:

DIRECCION INICIAL : ?

Aquí se debe ingresar una dirección hexadecimal comprendida entre las otras dos. Cualquier otra dirección será aceptada, con lo que se correrá el riesgo de alterar información del sistema operativo de la C-64, del editor y/o sus variables. No podrán ser ingresadas direcciones no hexadecimales o aquellas que no tengan cuatro (4) caracteres. En este caso el editor interrogará nuevamente sobre la dirección inicial.

Ejemplo de direcciones correctas
DIRECCION INICIAL : ? 200A
DIRECCION INICIAL : ? AFF9
DIRECCION INICIAL : ? 001A
Ejemplo de dirección incorrecta :
DIRECCION INICIAL : ? 200AB (#)
DIRECCION INICIAL : ? AA0 (#)
DIRECCION INICIAL : ? AGBB (#)
Nota (#) : El mensaje se volverá a imprimir nuevamente.

En el caso en que la dirección sea correcta, el PC tomará el valor de la misma y su valor se imprimirá debajo del mensaje anterior. Luego aparecerá al lado de ésta otro signo de interrogación el cual significa que el editor está esperando una instrucción o un dato correspondiente al valor actual del PC. Si se trata de ingresar valores de datos o instrucciones no hexadecimales, el editor volverá a imprimir el valor del PC sin incrementar, quedándose a la espera del nuevo dato. Además, si éste es mayor a los dos caracteres (hexadecimales) el editor procederá de igual manera. Es decir que el dato a ingresar debe tener uno o dos caracteres. En el primer caso se insertará un "0" delante del caracter ingresado. En estos casos, el PC se incrementará en uno, imprimirá su nuevo valor y esperará la entrada del dato siguiente.

Ejemplos:
DIRECCION INICIAL : ? 1B00 (cr)
1B00 ? AT (cr) dato erróneo. Se imprime PC nuevamente
1B00 ? AA (cr) dato correcto. Se incrementa PC en uno
1B01 ? BBB (cr) supera los dos carac. hexadecimales
1B01 ? F1 (cr) dato correcto. Se incrementa PC en uno
1B02 ? A (cr) dato correcto. Se asume 0A
1B03 ? A9 (cr) dato correcto. Se imprime PC+1
1B04 ? (cr) Se asume el dato anterior (A9)
1B05 ? (cr) Se asume el dato anterior (A9)
1B06 ? (cr) Se asume el dato anterior (A9)
1B07 ? editor a la espera
Nota: Este programa no es correcto. Sólo se utilizó para ejemplificar lo dicho.

Una vez que culminamos con el proceso de carga, podemos listar el programa en pantalla, ejecutarlo, modificarlo o borrar algunas direcciones. También podemos almacenarlo en el disco para

luego utilizarlo o, simplemente, finalizar nuestra tarea. Todo es posible conmutando el editor de modo carga a modo comando.

ii) Modo Comando:

La conmutación antes mencionada se realiza a través de un ".". Este punto se ingresa como si fuese un dato. De esta manera el editor entiende que el próximo caracter leído es un comando.

En nuestro ejemplo:

1B07 ? . (cr) se conmuta a modo comando

? el editor queda a la espera de un comando

En este modo el contenido del PC no se imprime. Sólo se imprime un signo de interrogación. De esta forma el editor nos indica su disponibilidad para recibir el comando.

Este puede ser:

CLASE I

N (New) ; R (Run) ; L (List) ; M (Modifica) ; D (Delete) ; C (cambia PC) S (Stop) ; B (Bye) ; X, Y, P, ✱(imprime el contenido de los registros X,Y,SP y acumulador respectivamente); ↑(Save) (Load) ←

CLASE II

@ ,(imprime el directorio del diskett actual)

@ v (envía comando VALIDATE al disk drive)

@ J (envía comando INITIALIZE al disk drive)

@ S (borra File deseado)

Comando New:

Función: Resetea el editor. Origina que se imprima el mensaje

DIRECCION INICIAL : ?

De esta forma se comienza nuevamente. Cabe resaltar que la información cargada no es destruída por el editor cuando este comando es solicitado. Es decir que el programa puede ser recuperado.

Formato: N

---()---

Comando Run:

Función: Produce la ejecución del programa almacenado desde la dirección inicial actual. Cuando el programa devuelva el control al editor, éste pasará a modo carga.

Formato: R

---()---

Comando List:

Función: Lista el contenido de la/s dirección/es deseadas.

Formato: Lcccc-ffff

Lcccc-

L-ffff

Ldddd

L

donde: cccc dirección de comienzo

ffff dirección de fin

dddd dirección deseada

El primer formato lista los contenidos de las direcciones hexadecimales comprendidas entre la dirección cccc y la dirección ffff incluída. Si la dirección cccc es mayor a ffff, el comando no se ejecuta y el editor vuelve al modo carga.

El segundo formato lista los contenidos de las direcciones hexadecimales comprendidas entre cccc y la dirección representada por (PC)-1 (contenido del PC menos uno). Si la dirección cccc es mayor que (PC)-1, se imprimirá, solamente, el contenido de la dirección cccc. Luego el editor pasará a modo carga.

El tercer formato lista los contenidos de las direcciones hexadecimales comprendidas entre la dirección inicial actual (como luego veremos tanto el PC como la dirección inicial pueden ser cambiados durante el desarrollo del programa), hasta la dirección ffff. En el caso en que ésta sea menor o igual a la dirección inicial, el editor imprimirá el contenido de la dirección inicial solamente. Luego volverá a modo carga.

El cuarto formato lista únicamente el contenido de la dirección hexadecimal representada por dddd.

Por último, el quinto formato lista los contenidos de las direcciones hexadecimales comprendidas entre la dirección inicial actual y la dirección representada por (PC)-1.

En todos los casos el listado puede interrumpirse con sólo presionar una tecla. Además, cuando finalice el comando, ya sea a través de una interrupción o por la finalización de él, el editor volverá al modo carga imprimiendo el contenido del PC actual juntamente con el signo de interrogación. En nuestro ejemplo, los formatos darían los siguientes resultados:

(Como se recordará, el editor había quedado a la espera de un dato o de un punto en la dirección 1B07 ((PC)=1B07)

1B07 ? . (cr)

? L (cr)

1B00 AA

1B01 F1

1B02 0A

1B03 A9

1B04 A9

1B05 A9

1B06 A9

1B07 ? . (cr) el editor vuelve a modo carga

? L1B02-1B05 (cr)

1B02 0A

1B03 A9

```
1 REM * ASEMBLER 1.1 CRISTIAN J PARODI *
3 REM *** PANTALLA E INICIALIZACION ***
4 POKE53280,7:POKE53281,6:GOSUB1000
5 P=0:SA=0:CU=0:F=0:AD=0:T=0:I=0:M=0:N=0:CA=0
  :R=0:C=0
7 H$="":P$="":CO$="":M$=""
9 REM ******** MOD PRINCIPAL ********         *
  **** ELECCION DE COMANDO *****
10 PRINT"□":INPUT"            ■DIRECCION INICIAL
  :■ ";A$
11 IFLEN(A$)<>4THEN10
12 GOSUB500:IFF=1THENF=0:GOTO10
14 PRINT
20 CA=P:SA=P:AD=P
30 SYS49152:N=AD:GOSUB700:PRINTTAB(5);P$:INPUT"
   ▶▶▶▶▶▶▶▶▶▶▶";A$:IFA$<>"."THEN53
31 PRINT:INPUT"▶▶▶▶▶▶▶▶▶▶▶";A$
32 IFA$="*"THENSA=780:CU=780:F=1:GOSUB800:GOTO30
33 IFA$="X"THENSA=781:CU=781:F=1:GOSUB800:GOTO30
34 IFA$="Y"THENSA=782:CU=782:F=1:GOSUB800:GOTO30
35 IFA$="P"THENSA=783:CU=783:F=1:GOSUB800:GOTO30
36 IFMID$(A$,1,1)="B"THENSYS58260
37 IFMID$(A$,1,1)="R"THENSYSCA:GOTO30
38 IFMID$(A$,1,1)="C"THENGOSUB250:GOTO30
39 IFA$="@"THENGOSUB750:GOTO30
40 IFMID$(A$,1,2)="@S"THENGOSUB150:GOTO30
41 IFMID$(A$,1,2)="@V"THENGOSUB170:GOTO30
42 IFMID$(A$,1,2)="@I"THENGOSUB178:GOTO30
43 IFMID$(A$,1,1)="M"THENGOSUB100:GOTO30
46 IFMID$(A$,1,1)="D"THENGOSUB200:GOTO30
47 IFMID$(A$,1,1)="L"THENGOSUB290:GOTO30
48 IFMID$(A$,1,1)="S"THENSTOP:GOTO30
49 IFMID$(A$,1,1)="↑"THENGOSUB400:GOTO30
50 IFMID$(A$,1,1)="←"THENGOSUB600:GOTO30
51 IFMID$(A$,1,1)="N"THEN5
52 PRINTTAB(5);"COMANDO DESCONOCIDO":PRINT:GOTO30
53 IFLEN(A$)>2ORLEN(A$)=0THEN30
58 GOSUB500:IFF=1THENF=0:GOTO30
60 POKEAD,P:AD=AD+1:GOTO30
99 REM *** MODIFICACION DEL ADDRESS ***
100 IFLEN(A$)<>8THENGOSUB1010:RETURN
105 M$=MID$(A$,2,4):CO$=MID$(A$,7,2):A$=M$:
    GOSUB500:IFF=1THENGOSUB1010:RETURN
110 M=P:A$=CO$:GOSUB500:IFF=1THENGOSUB1010:
    RETURN
111 N=PEEK(M):POKEM,P:GOSUB700
```

```
1B04   A9
1B05   A9
1B07 ? . (cr)
       ? L-1B03 (cr)
1B00   AA
1B01   F1
1B02   0A
1B03   A9
```

```
1B07 ? . (cr)
       ? L1B04 (cr) lista el contenido de
         la direc. 1B04
1B04   A9
1B07 ? . (cr)
       ? L1B03- (cr)
1B03   A9
1B04   A9
```

```
1B05   A9
1B06   A9
1B07 ? . (cr)
       L1B06-1B02 (cr) la direc. cccc
       es mayor que la ffff. el comando
       no se ejecuta.
1B07 ? . (cr)
       ? L-1AFE (cr) la dirección ffff es
```

```
115 PRINTTAB(5);"■ DATO ANTERIOR :■ ";P$
116 PRINTTAB(5);"■ DATO ACTUAL   :■ ";CO$
120 PRINT:RETURN
148 REM *** BORRA FILE DESEADO ***
150 INPUT"      NOMBRE DEL FILE:";FL$
152 IFFL$="."THENPRINT:RETURN
154 IFLEN(FL$)>8THEN150
156 OPEN15,8,15,"S:"+FL$
158 PRINTTAB(5);"■FILE■ ";FL$;" ■BORRADO■"
160 CLOSE15:PRINT:RETURN
168 REM *** COMANDO VALIDATE ***
170 OPEN15,8,15,"V"
172 PRINTTAB(5);"■REORGANIZANDO EL DIRECTORIO■"
    :PRINT
174 CLOSE15:RETURN
176 REM *** COMANDO INITIALIZE ***
178 OPEN15,8,15,"I"
180 PRINTTAB(5);"■INICIALIZANDO DISK DRIVE■":
    PRINT
182 CLOSE15:RETURN
198 REM *** PONE NOP EN EL ADDRESS ***
200 IFLEN(A$)<>5THENGOSUB1010:RETURN
205 A$=MID$(A$,2,4):GOSUB500:IFF=1THENGOSUB1010
    :RETURN
206 POKEP,234
208 PRINTTAB(5);"■DIRECCION■ ";A$;" ■BORRADA■"
    :PRINT
210 RETURN
248 REM *** CAMBIO DE PC ***
250 IFLEN(A$)<>5THENGOSUB1010:RETURN
254 A$=MID$(A$,2,4):GOSUB500
258 IFF=1THENGOSUB1010:RETURN
259 AD=P:M$=""
260 INPUT"■■■■ NUEVA DIRECCION INICIAL :";M$
261 IFM$=""THENRETURN
262 IFLEN(M$)<>4THEN260
264 A$=M$:GOSUB500:IFF=1THEN260
268 CA=P:RETURN
288 REM *** SUBPGM COMANDO LIST ***
290 IFLEN(A$)<>1ANDLEN(A$)<>6ANDLEN(A$)<>
    10ANDLEN(A$)<>5THENGOSUB1010:RETURN
300 IFMID$(A$,2,8)=""THENSA=CA:CU=AD-1:GOSUB800
    :RETURN
310 IFMID$(A$,2,1)="-"ANDMID$(A$,3,4)<>""
    THENGOSUB850:RETURN
320 IFMID$(A$,2,4)<>""ANDMID$(A$,6,1)="-
    "ANDMID$(A$,7,8)=""THENGOSUB900:RETURN
330 IFMID$(A$,2,4)<>""ANDMID$(A$,6,8)=""
    THENGOSUB950:RETURN
340 GOSUB980:RETURN
398 REM *** COMANDO SAVE ***
400 INPUT"      NOMBRE DEL FILE:";FL$
402 IFSA=ADTHENI=0:GOTO404
403 I=1
404 IFLEN(FL$)>8THEN400
410 OPEN3,8,3,"@0:"+FL$+",S,W"
411 CU=AD:SA=CA:PRINT#3,SA:PRINT#3,CU
412 PRINT"      ■SAVING :■";FL$
413 N=SA:GOSUB700:PRINT"      ■DIRECCION INICIAL
    :■ ";P$
414 N=AD-I:GOSUB700:PRINT"      ■DIRECCION FINAL
    :■ ";P$
415 FORT=SATOCU:N=PEEK(T)
416 PRINT#3,N
418 NEXTT:PRINT
420 CLOSE3:RETURN
490 REM *** HEX-->DEC ***
500 T=LEN(A$)-1:P=0
510 FORI=1TOLEN(A$)
520 H$=MID$(A$,I,1)
530 IFASC(H$)>=48ANDASC(H$)<=57THENF=48:GOTO550
540 IFASC(H$)>=65ANDASC(H$)<=70THENF=55:GOTO550
542 F=1:RETURN
550 P=P+(ASC(H$)-F)*16↑T:T=T-1
560 NEXTI:F=0:RETURN
598 REM *** COMANDO LOAD ***
600 INPUT"      NOMBRE DEL FILE:";FL$
604 IFLEN(FL$)>8THEN600
610 OPEN15,8,15:OPEN3,8,3,"@0:"+FL$+",S,R"
611 GOSUB650:IFF=1THENF=0:CLOSE3:CLOSE15:RETURN
612 INPUT#3,SA:INPUT#3,CU
614 PRINT"      ■LOADING :■ ";FL$.
616 N=SA:GOSUB700:PRINT"      ■DIRECCION INICIAL
    :■ ";P$
618 N=CU-1:GOSUB700:PRINT"      ■DIRECCION FINAL
    :■ ";P$
620 FORT=SATOCU
622 INPUT#3,N:POKET,N
624 NEXTT:PRINT
626 CLOSE3:CLOSE15:RETURN
648 REM *** READ CANAL DE COMANDOS ***
650 INPUT#15,I,M$,C,R
652 IFI=62THENF=1:PRINT"■";TAB(12);"■"M$"■":
    PRINT:RETURN
654 RETURN
690 REM *** DEC-->HEX ***
700 H$="":P$="":IFA$=""THENRETURN
710 C=INT(N/16)
712 R=N-16*C
714 IFR>=0ANDR<=9THENF=48:GOTO718
716 F=55
718 H$=H$+CHR$(R+F)
720 IFM=1THENM=0:GOTO724
721 IFC<16THENR=C:M=1:GOTO714
722 N=C:GOTO710
724 FORT=LEN(H$)TO1STEP-1
726 P$=P$+MID$(H$,T,1):NEXTT:F=0:IFLEN(P$)>4
    THENP$="00":AD=0
728 RETURN
748 REM *** DIRECTORIO ***
750 OPEN15,8,15:OPEN1,8,0,"$0"
752 GET#1,M$,H$
754 GET#1,M$,H$:GET#1,M$,H$
757 IFM$<>""THENC=ASC(M$)
758 IFH$<>""THENC=C+ASC(H$)*256
759 PRINTTAB(7);MID$(STR$(C),2);TAB(11);
```

menor que la direc. inicial.
1B00 AA
1B07 ? .
? L1B11- (cr) la direc. cccc es mayor que (PC)-1
1BAA FF
1B07 ? . (cr)
? L2- (cr)

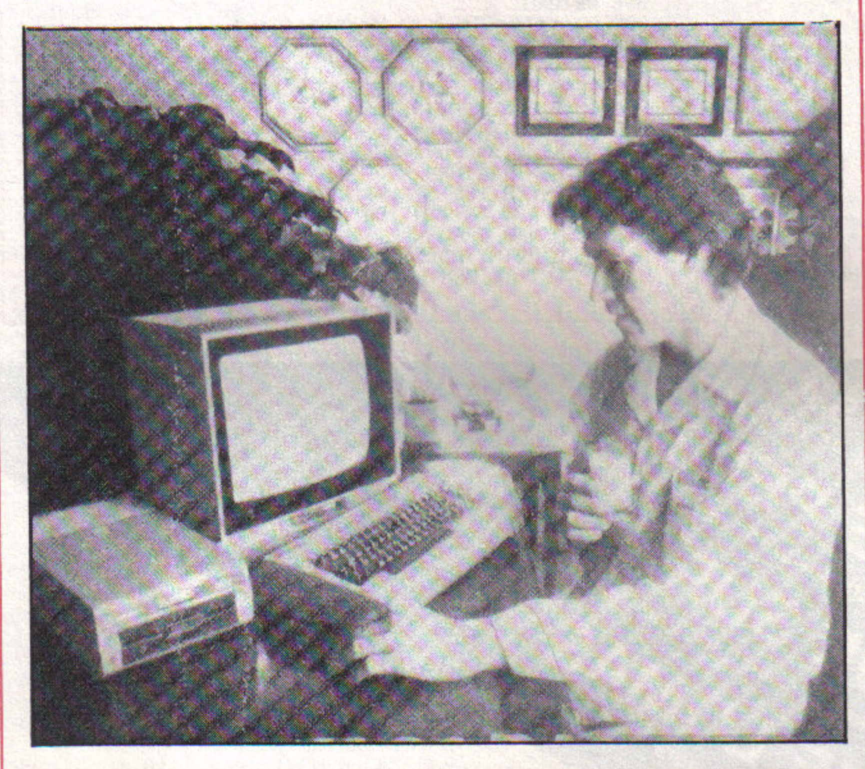
ERROR DE COMANDO
1B07 ? . (cr)
? L0002- (cr) listará los contenidos de las direc. comprendidas entre 0002 (hexadecimal) y la 1B06. Cuando finalice el editor vilverá a modo carga

1B07 ?
Se debe tener cuidado con la sintaxis de los comandos. Debe respetarse los formatos establecidos. En caso que los formatos fuesen erróneos, el editor imprimirá el mensaje:
ERROR DE COMANDO
y volverá a modo carga.

```
760 GET#1,H$:IFST<>0THEN769
761 IFH$<>CHR$(34)THEN760
762 GET#1,H$:IFH$<>CHR$(34)THENPRINT"▯"H$"■";
    :GOTO762
763 GET#1,H$:IFH$=CHR$(32)THEN763
764 PRINTTAB(28);:CO$=""
765 CO$=CO$+H$:GET#1,H$:IFH$<>""THEN765
766 PRINTTAB(8);LEFT$(CO$,3)
767 REM * PRUEBA FIN DE LECTURA *
768 IFST=0THEN754
769 PRINT"▯BLOCKS FREE■";:C=0:PRINT"▯"
770 CLOSE1:CLOSE15:RETURN
798 REM *** BLOQUE LISTADOR ***
800 FORI=SATOCU:SYS49152:GETM$:IFM$<>""THENRETURN
801 N=PEEK(I)
802 IFN=0ANDF=1THENF=0:PRINTTAB(12);"00":RETURN
803 IFN=0ANDF=0THENCO$="00":GOTO807
804 IFN<>0ANDF=1THENF=0:GOSUB700:PRINTTAB(12);
    P$:RETURN
805 GOSUB700:CO$=P$
807 N=I:GOSUB700:M$=P$
808 PRINTTAB(5);M$;TAB(12);CO$:NEXTI
809 RETURN
850 A$=MID$(A$,3,4):GOSUB500:IFF=1THENGOSUB1010
    :RETURN
854 CU=P:SA=CA:GOSUB800
856 RETURN
900 A$=MID$(A$,2,4):GOSUB500:IFF=1THENGOSUB1010
    :RETURN
904 SA=P:CU=AD-1:GOSUB800
906 RETURN
950 A$=MID$(A$,2,4):GOSUB500:IFF=1THENGOSUB1010
    :RETURN
954 SA=P:CU=P:GOSUB800
956 RETURN
980 CO$=MID$(A$,2,4):M$=MID$(A$,7,4)
982 IFCO$>M$THENPRINT:RETURN
984 A$=CO$:GOSUB500:IFF=1THENGOSUB1010:RETURN
986 SA=P:A$=M$:GOSUB500:IFF=1THENGOSUB1010:RETURN
988 CU=P:GOSUB800:RETURN
998 REM *** SYS ***
1000 POKE2,194:FORI=49152TO49169:READC:POKEI,
     C:NEXT:RETURN
1002 DATA 72,165,52,197,2,16,9,169,255,133,51,169,
     202,133,52,234,104,96
1005 REM *** ERROR DE SINTAXIS ***
1010 F=0:PRINTTAB(5);"ERROR DE COMANDO":
     PRINT:RETURN
```

INSTRUCCIONES PROPIAS DE COMMODORE

Quienes hayan usado más de un microcomputador, habrán comprobado que el Basic es un lenguaje que intenta ser standard. Pero cada fabricante trata de implementar una versión más ventajosa que las demás. Por eso en esta nota nos proponemos detallar las instrucciones particulares del equipo C-64



Sentencias FRE

Función: permite determinar la capacidad de memoria libre del microcomputador.
Modo de empleo: Cargar el siguiente programa y darle RUN, arrojará en K, la memoria disponible.

```
10 A=FRE (1)
20 If A> = 0 THEN 40
30 A = 65535 + A
40 A = A/1024
50 PRINT "MEMORIA LIBRE"; A
60 END
```

Comando CONT

Función: permite continuar la ejecución de un programa interrumpido mediante la instrucción o comando STOP.
Formato:
CONT return

Sentencia PEEK

Función: tomar el valor contenido en una localización de memoria determinada. Su valor será entre 0 y 255
Formato:
Ejemplo: A = PEEK (53281)

Sentencia POKE

Función: permite a un área o localización de memoria, darle un valor que indicará el estado de dicha localización
Formato:
POKE localización de memoria, valor
El valor será entre 0 y 255
Ej.: POKE 53280,0

SYS

Función: permite llamar y ejecutar una rutina escrita en lenguaje de máquina.
Formato:
SYS n
El valor de n será entre 0 y 65535. El varlo de n indicará la posición de memoria a partir de la cual se comenzó a cargar la rutina de máquina

ST (STATUS)

Función: permite mediante una señal indicar el fin de un archivo secuencial
Formato: El valor de ST puede ser 0 ó 64. Si es 64 indica fin de archivo, caso contrario indica habilitación para continuar con la lectura.
Ej.

```
100 IF ST = 64 THEN PRINT
"CERRAR CANAL": END
110 IF ST = 0 THEN PRINT
"LEER ARCHIVO": END
```

WAIT

Función: permite crear intervalos de tiempo mientas se ejecuta un programa. El lapso está dado por el cambio de estado de una posición de memoria determinado.
Formato:
WAIT, posición de memoria, nl, n2
nl y n2 son valores que oscilan entre 0 y 255

USR

Función: sirve para, desde un programa BASIC, saltar a un programa en lenguaje de máquina, cuyo punto de partida está en las posiciones de memoria 1 y 2.
Formato:
USR (posición de memoria)

TI - TI$

Función: sirve para controlar el reloj del computador. Cuando se enciende el equipo, el reloj arroja valor 0.
Formato:
TI$ = "000000"

A: SUPERIOR. B: MUY BUENO. C: BUENO. D: REGULAR. E: MALO. F: INACEPTABLE.

CABECERA DE PLAYA

Rating Total: A–
Creatividad: B+
Documentación: B
Profundidad de juego: B
Desafío: Complejo
Gráficos: A
Valor en relación al costo: B+
Mantiene el interés: A –



Es un juego de acción para uno o dos jugadores, a todo color, con una gran resolución de gráficos en tres dimensiones y excelentes efectos de sonidos que se pueden modificar y ajustar a gusto.
El objetivo del juego es capturar la fortaleza enemiga de Kuhn-Lin penetrando las defensas de la isla. Hay dos estrategias de ataque: un asalto total frontal contra una flota fuerte y preparada, seguido de un desembarco de playa; o un intento de entrar a escondidas a través de un pasaje secreto minado, y tomar a la flota enemiga por sorpresa. El acceso por el pasaje es lo más recomendable, aunque cada barco debe navegar cuidadosamente por un estrecho pasaje plagado de minas flotantes y velocísimos torpedos lanzados desde la costa. El jugador puede perder alguno de sus barcos pero a la larga, la sorpresa puede ser la única llave de la victoria.
Una vez producido el contacto, inmediatamente el enemigo lanza un bombardeo aéreo contra la flota. La única defensa es un cañón antiaéreo con elevación y dirección ajustables. Tiene cargas de munición ilimitadas, pero los encargados de abastecerlo trabajan más despacio que los que disparan. La puntería es importante porque se deberá averiar a los aviones en sus puntos más vulnerables para registrar una baja a favor. Si se es lento los aviones dejarán caer sus bombas que podrán dañar o hundir la flota. El ataque es mucho más rápido si se toma el pasaje secreto.
Después del ataque aéreo se podrá hundir a la flota enemiga con la artillería. Los barcos enemigos dispararán a distancia cada tanto, tratando de hacer blanco en las naves. Se deberá eliminarlos disparando y ajustando los cañones hasta que se encuentre el alcance correcto. Las salpicaduras frente a los buques enemigos indican que se ha quedado corto en el intento y, presumiblemente, aquellos que no provoquen salpicaduras, significan que siguieron de largo. Puede parecer simple, pero muchas veces, hasta los ajustes más sutiles en tiros por elevación podrán cumplir el cometido. Cuando se elimine la flota enemiga, aún no se habrá ganado la guerra. La parte más difícil todavía no llegó.
El obstáculo final es la fortaleza en sí misma. Se obtienen dos tanques por cada barco que sobrevivió a la primera batalla hasta un tope de ocho tanques. La meta es destruír los diez objetivos que se encuentran sobre un lado de la fortaleza.
Nuevamente la velocidad es esencial ya que la fortaleza está protegida por un cañón que nunca falla.
"Cabecera de playa" presenta buenos gráficos y efectos de sonido. Durante el ataque aéreo los aviones vuelan hacia uno con increíble realismo desapareciendo y apareciendo alternativamente. Uno deseará esquivarlos cuando los vea venir.
Cuando la batalla naval está en apogeo, los disparos de la artillería pesada que son errados caen en el océano salpicando agua.
Se puede hacer una pausa o parar el juego en cualquier momento, lo que permite dejar el control de la computadora y que ella juegue por sí sola hasta que se vuelva a usar el joystick otra vez. De esta forma se puede empezar a jugar en cualquier momento del juego y practicar en ciertas secuencias.
Al tener cuatro niveles de dificultad, el juego garantiza que, incluso, el más fanático de los juegos de puntería no se aburra, porque los niveles más avanzados son rápidos y violentos.

BLUE MAX

Rating Total: A
Creatividad: B+
Profundidad de juego: B+
Desafío: Complejo - Gráficos: A
Valor en relación al costo: A–
Mantiene el interés: A–



Es un juego de combate aéreo en tres dimensiones. La misión es destruír la mayor cantidad posible de aviones enemigos y al mismo tiempo hacer volar los objetivos enemigos en tierra bombardeando o ametrallándolos en vuelos rasantes. Eventualmente se deberá llegar a la ciudad en la que habrá que bombardear tres objetivos específicos para ganar.
La aviación se controla con el joystick. Por alguna extraña razón el autor ha instalado los controles en la parte de atrás. De todas maneras si se seleccionan los controles invertidos el juego andará bien.
Una vez en la posición de control invertida se puede tirar hacia atrás el joystick para hacer ascender el avión, y empujar hacia adelante para hacerlo descender. La ametralladora se acciona con un botón disparador y las bombas son lanzadas al apretar el disparador mientras se desciende perdiendo altitud.
En un tablero instrumental puede leerse la velocidad, el combustible, la altitud y el estado del equipo del avión. Cuando hay una avería, una señal luminosa lo indicará. Del mismo modo, un destello en un asterisco indicará si algún avión enemigo se encuentra encima o debajo de la nave o si se acerca por delante o por detrás.
El avión despega de una pequeña pista. Debe alcanzar las 100 millas por hora antes de levantar vuelo o si no se estrellará. Una vez en el aire comenzará a sobrevolar el terreno enemigo que incluye un río fuertemente fortificado

con artillería antiaérea y algunas barcazas móviles. El avión proyecta una sombra sobre el ondulado terreno dando una buena indicación visual de la altitud y proveyendo una buena vista del sitio a bombardear. El viento juega su parte en el juego ya que las bombas al caer tienden a ser desviadas por su fuerza. Algunos de los blancos marcados con una X ofrecen mayor puntaje si se las destruye. Las ametralladoras del avión pueden ser usadas para "bombardear" desde altitudes de entre ventiún y veinticinco pies.

Cuando se ha marcado 1000 puntos la pantalla diseña una ruta y el campo de vuelo del enemigo. Se puede bombardear el hangar y varios aviones estacionados al lado de la pista. Luego el terreno comienza a ondularse hacia el río. Sólo después de que se ha alcanzado un puntaje de 5000 la pantalla mostrará la ciudad del enemigo. El objetivo aquí es bombardear tres objetivos especiales en el centro de la ciudad. Se deberá volar entre altos edificios para llegar a los blancos.

La parte más excitante de Blue Max se encuentra en las batallas aéreas. Los aviones se acercan por delante o por detrás. Se debe intentar volar a la misma altura y apuntar las ametralladoras. Porque averiar a un avión que se halle encima es muy peligroso ya que es más fácil chocar con él que derribarlo. Los aviones que vienen por detrás son más fáciles de atacar si se espera que pasen y luego se dispara.

Mientras los aviones vuelan a través de la línea de fuego antiaérea es muy posible que sufran algún daño. Los indicadores luminosos del tablero advertirán qué clase de daño han sufrido: menor maniobrabilidad, pérdida de combustible, o desperfecto en el sistema de bombardeo o de ametralladoras. Cuando los cuatro indicadores titilan el avión se estrellará. Se puede arreglar el avión recargando combustible y reparándolo en el aeropuerto. La misma computadora alertará cuando se esté acercando al lugar de reparaciones. Se deberá bajar el tren de aterrizaje, volar despacio y aterrizar lo más cerca posible del edificio central. Si se hace demasiado lejos de la pista nunca se va a poder alcanzar la velocidad de despegue, antes de llegar al fin de la pista. Además, se convertirá en un blanco perfecto, en la pista, ya que los aviones enemigos estarán sobrevolando.

Blue Max es un juego muy divertido que da la sensación realista de estar volando un bi-plano. Es un entretenimiento desafiante que implica, no obstante cierta práctica para no caer derrotado. Como hay un solo avión en el comando, si se estrella, estará obligado a comenzar el juego nuevamente. Blue Max tiene una gran profundidad en relación al juego y mantiene el interés de casi todos por grandes períodos.

DONKEY KONG



Rating total: A
Creatividad: A
Documentación: B
Profundidad del juego: A
Desafío: Mediano
Gráficos: A
Valor en relación al costo: A
Mantiene el interés: A

El accionar del valiente Mario al ponerse en acción para rescatar a su novia de las garras del gorila Donkey Kong (que la mantiene cautiva en la cima de una pila de vigas rotas), ha convertido a "Donkey Kong" en uno de los más famosos juegos de computadoras.

Mario comienza su intento de rescate al pie de una pila de vigas inclinadas de una obra en construcción. Algunas de estas vigas están conectadas por escaleras. Donkey Kong, un enorme y siniestro gorila, intenta impedir a Mario que trepe para rescatar a su amada, arrojándole barriles a través de rampas. Mario debe saltarlos o romperlos con un martillo a medida que vienen hacia él. Es preferible ignorar este segundo método y proseguir tan rápido como sea posile hacia la cima ya que los premios disminuyen a medida que va pasando el tiempo.

Mario se mueve a través del joystick y para hacerlo saltar se debe oprimir el botón. Se debe tener cuidado con los barriles que van cayendo porque algunas veces acortan su camino bajando por una escalera. Los barriles que llegan a la base son incendiados en una lata que los transforma en bolas de fuego que pueden alcanzar a Mario si éste no corre rápidamente hacia arriba. Para dominar este nivel de juego hay que hacer buen tiempo y poseer destreza.

Mario pasa a un segundo tramo del juego en el que se presenta una pirámide

de vigas. Su meta es destruir los ocho remaches de acero que mantienen unida la estructura, al tiempo que debe evitar caer por las brechas y esquivar las bolas de fuego. Esta parte del juego no ofrece gran dificultad. Si Mario se ecuentra acorralado podrá utilizar un martillo que está próximo a la cúspide para defenderse de las bolas de fuego. No es necesario llegar a todos los remaches y anotarse todos los puntos que deje caer la novia aún prisionera.

La tercera parte es quizás la más ardua. Consiste en una serie de plataformas conectadas por escaleras y separadas por dos ascensores ligeros. Numerosas bolas de fuego rodean las plataformas y complican el paso de una a otra. Donkey Kong arroja vigorosamente pesadas vigas desde lo alto. Mario debe calcular con mucha precisión para poder saltar entre esta interminable catarata de vigas. Este nivel requiere mucha más habilidad y destreza que los demás.

La última parte es similar a la segunda. Consiste en una complicada serie de cintas sinfin conectadas por escaleras. Las bolas de fuego persiguen a Mario mientras él intenta saltar arenas movedizas que se hallan en las cintas sinfin. El objetivo sigue siendo el rescate de la amada.

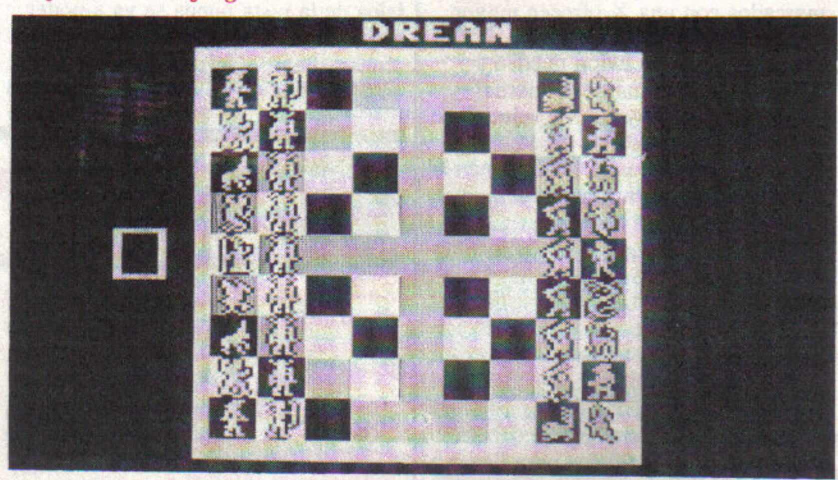
Los efectos gráficos y de sonido son excelentes. Todos los personajes están cuidadosamente dibujados con colores brillantes y muy buena animación. Las bolas de fuego flamean igual que las reales y los barriles dan la exacta sensación de estar volando.

El juego puede ser jugado por una o dos personas en turnos alternativos. Se puede comenzar por diferentes niveles pero siempre a partir de la primera parte descripta.

Cada jugador comienza el juego con tres vidas y gana una vida extra al alcanzar los 7000 puntos. Donkey Kong, para la Commodore 64, es casi tan bueno como la popular versión original.

ARCHON

Rating Total: A
Creatividad: A
Documentación: B+
Profundidad del juego: A
Desafío: Complejo
Gráficos: B+
Valor en relación al costo: A
Mantiene el interés: A



Este juego recuerda esas escenas humorísticas de la "Guerra de las Galaxias" en las que Arturito (R2-D2) y Chewbaca jugaban a una especie de ajedrez a bordo de la Millennium Falcon.

Es un tablero estratégico similar al de ajedrez y hay una serie de piezas que parecen personajes de fantasía. El objetivo del juego es derrotar a cada una de las piezas enemigas en un combate, o controlar los cinco puntos del poder.

Igual que en el ajedrez cada pieza tiene cierta fortaleza, velocidad y movimientos específicos. Pero a diferencia del ajedrez los contrincantes, la Luz y la Oscuridad, tienen piezas que difieren unas de otras aunque en conjunto tienen el mismo poder.

Cada pieza del tablero tendrá una contrapartida con características similares pero diferente estilo de lucha. Por ejemplo, el equivalente al peón de ajedrez, es un caballero para las fuerzas de La Luz y tienen un mago, dos unicornios, dos arqueros, dos Golems (forma artificial de piedra y metal con forma de hombre), dos walkirias, un Djinni (ser mágico proveniente de otra dimensión), un Phoenix, y siete caballeros. Los Golens arrojan piedras mientras que el Djinni lanza tornados y el Unicornio dispara proyectiles de energía.

Las fuerzas de La Oscuridad tienen una Hechicera, dos basilisk (semejantes a un reptil de ojos saltones), dos manticores (leones de oro con cara humana y cola de escorpión), dos trolls (espíritus de la oscuridad), un

shapeshifter (que no tiene un contorno determinado un puede tomar la forma de su oponente), un dragón, dos hanshees (espíritus que se alimentan de espíritus) y siete duendes. Los basiliks atacan con rayos que disparan desde sus ojos, los manticores, con rayos de sus colas, etc.
Tanto el Mago de La Luz como la Hechicera de La Oscuridad pueden emitir hechizos además de pelear con bolas de fuego y flechas luminosas respectivamente.
El campo de batalla es diferente del tablero estratégico. Cuando dos adversarios se enfrentan ocupando los dos el mismo casillero, ya sea de La Luz, La Oscuridad o neutral, luchan en la arena de combate que presenta barreras que lo llevan a través de un cielo de luminosidad. Cuando las barreras son del mismo color que el fondo (los dos, el de La Luz o La Oscuridad) desaparecen.
Durante unos segundos antes y después de este momento, los oponentes pueden pasar sobre ella o penetrarla con un misil, aunque las barreras retardarán la velocidad del mismo.
Mientras tanto, las líneas de vida de los costados del tablero indican la potencia relativa de cada oponente.
Si algún personaje es herido, la línea se reduce en proporción a la herida recibida. Cuando la línea baja a cero, el enemigo está muerto y el combate termina.
El combate es una mezcla de estrategia y destreza. Un duende débil puede a veces derrotar a un poderoso unicornio si consigue arrinconarlo y golpearlo en la cabeza sin ser alcanzado por el primer proyectil que lance el equino. Durante el combate los gráficos presentan una excelente animación.
Tanto el Mago como la Hechicera son dos figuras poderosas que no deben ser sacadas de sus puntos de poder. Estos casilleros son importantes ya que en ellos se acelera la curación de los heridos en combate.
La magia es el elemento más importante dentro de la estrategia del juego. Hay siete clases de hechizos y cada uno puede ser usado sólo una vez. Algunos de los más interesantes son los del teletransportador, por el cual una pieza puede ser trasladada de un lugar a otro del tablero, cambiar hasta dos piezas del tablero estratégico, revivir algún personaje muerto, apresar algún enemigo, o convocar algún espíritu clemente (este último conjuro es particularmente bueno para atacar a un enemigo poderoso).
El hechizo del aprisionamiento es útil para mantener una figura enemiga en una determinada casilla. Durante el combate en los casilleros claros las fuerzas de La Luz tienen ventajas y en los oscuros la tienen los de La Oscuridad. Muchos de los casilleros (especialmente los neutrales) cambian de claros a oscuros siguiendo un parámetro prefijado. Otro hechizo llamado desplazamiento del tiempo revertirá el flujo del ciclo luminoso.
Archon es uno de los juegos más creativos y originales de los últimos años. Tiene sus raíces en el ajedrez y en los juegos fantásticos. Esta combinación funciona para lograr que esta conjunción se transforme en un excelente programa de estrategia y combate. Despierta el interés tanto de adolescentes como de adultos independientemente de los gustos o no por el ajedrez. A pesar de su complejidad, puede ser rápidamente asimilado por personas de todas las edades. Archon tiene muy buenos gráficos y puede hacer pasar excelentes momentos.

Con este número iniciamos una sección que servirá para que los lectores planteen sus consultas y sugerencias. Para eso deberán escribir a nuestra redacción cuya dirección es Cerrito 1320, 1º Piso Buenos Aires (1010).

Memoria Libre

- Cómo sé la memoria libre que dispongo en la consola?

Sencillamente debemos tipear:

PRINT FRE(1)

Si el valor es negativo debemos sumarle el valor 65535. Si recién encendimos la consola, el resultado de esta operación arrojará alrededor de 38000 que son los 38 K que dispone el BASIC.

Si el valor es positivo, es el valor libre en bytes que se dispone. Si deseamos obtenerlo en Kbytes, el resultado debemos dividirlo por 1024 (Recordemos que un kbyte es igual a 1024 bytes).

Dirección

- ¿Cómo encontramos la dirección de fin y comienzo de un programa en lenguaje de máquina almacenado en diskette?

Los primeros dos bytes de un archivo o programa en disco contienen el byte de más alto y bajo valor. Este programa dado a continuación encuentra el comienzo de la dirección y el fin es igual al comienzo más el número de bytes leídos.

```
10  INPUT "NOMBRE DEL
    PROGRAMA"; f$
20  A$ = "0:" + F$ + ",P,R"
30  OPEN 2,8,2, A$
40  GET#2, A$
50  GOSUB 150
60  L = A
70  GOSUB 150
80  SA = L + 256 * A
90  PRINT "DIRECCION
    COMIENZO "; SA
100 GET#2,A$
110 IF ST = 0 THEN SA = SA + 1:
    GOTO 100
120 PRINT "DIRECCION FINAL
    "; SA
130 CLOSE 2
140 END
150 IF A$ = "" THEN A$ =
    CHR$(0)
160 A$=ASC(A$): RETURN
```

Tiempo

- ¿Cómo se puede manejar el tiempo en un juego?

La sentencia que maneja el tiempo es TI. Esta variable se usa con el reloj de tiempo real incorporado a la C-64. Esta variable es actualizada cada 1/60 de segundo. Empieza en cero cuando se conecta el equipo y puede volverse a cero cambiando el valor de otra variable reservada TI$.

El TI$ es una cadena actualizada constantemente por el sistema operativo. Su formato es de 6 caracteres:

HHMMSS

(Horas, minutos y segundos)

A esta variable se le puede adjudicar cualquier valor (todos deben ser numéricos) y se mantendrá actualizada a partir de ese punto.

Ej. TI$ = "020001" return
(hora 2, 0 minuto, 1 segundo)

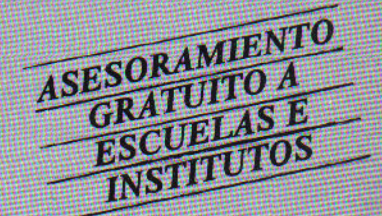


Drean commodore

AGENTE AUTORIZADO



ENVIOS AL
INTERIOR
C/GARANTIA
DE DREAN



Disponemos de:

JOYSTICKS - BIBLIOGRAFIA - DISKETTES
INTERFACES - ACCESORIOS - GRABADORES

En Breve:

DISKETTERAS - IMPRESORAS Y DATASETE



No damos precios por teléfono

DISPONEMOS DE UN AMPLIO STOCK
DE SOFTWARE ORIGINAL C/GARANTIA



S. A.

Av. CORRIENTES 2198 Cap. Tel. 46-2529/7877